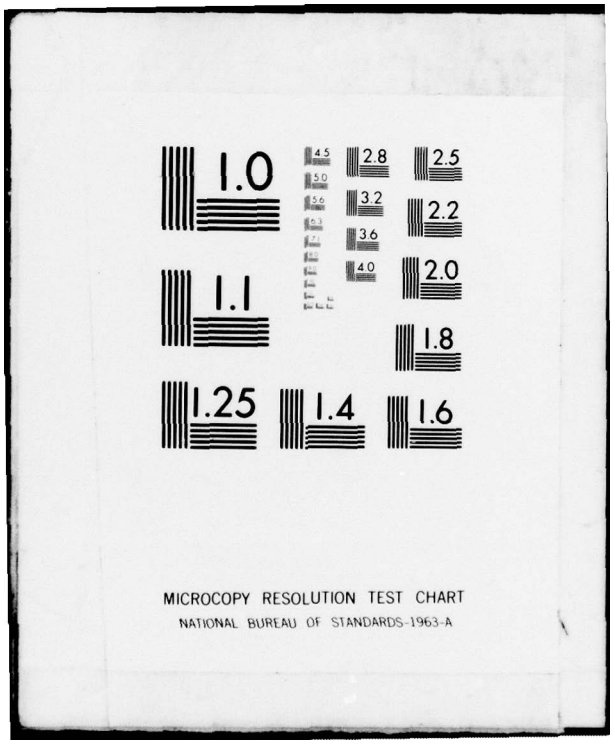
AD-A081 012 YALE UNIV NEW HAVEN CONN DEPT OF COMPUTER SCIENCE F/G 5/7
ADAPTIVE UNDERSTANDING: CORRECTING ERRONEOUS INFERENCES.(U)
JAN 80 R H GRANGER N00014-75-C-1111
UNCLASSIFIED RR-171 NL

1 OF 3
AD A081012


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

ADAPTIVE UNDERSTANDING: CORRECTING ERRONEOUS INFERENCES

January 1980

Research Report #171

Richard Horace Granger, Jr.

YALE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

This work was presented to the Graduate School of Yale University in candidacy for the degree of Doctor of Philosophy. The author is presently a member of the Department of Information and Computer Science at the University of California at Irvine.

ADAPTIVE UNDERSTANDING: CORRECTING ERRONEOUS INFERENCES

January 1980

Research Report #171

Richard Horace Granger, Jr.

This work was supported in part by the Advanced Research Projects Agency of the Department of Defense and monitored by the Office of Naval Research under contract N00014-75-C-1111.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: #171
2. GOVT ACCESSION NO.:
3. RECIPIENT'S CATALOG NUMBER:

4. TITLE (and Subtitle): (6) Adaptive Understanding: Correcting Erroneous Inferences.
5. TYPE OF REPORT & PERIOD COVERED: (9) Research Report.
6. PERFORMING ORG. REPORT NUMBER:

7. AUTHOR(s): (10) Richard Horace Granger, Jr.
8. CONTRACT OR GRANT NUMBER(s): (15) N00014-75-C-1111

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Yale University - Department of Computer Science
10 Hillhouse Avenue - P.O. Box 2158
New Haven, Connecticut 06520

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: (12) 196

11. CONTROLLING OFFICE NAME AND ADDRESS: (11)
Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, Virginia 22209

12. REPORT DATE: January 1980
13. NUMBER OF PAGES: 180

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):
Office of Naval Research
Information Systems Program
Arlington, Virginia 22217

15. SECURITY CLASS. (of this report): unclassified
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this report is unlimited. (14) RR-171

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Natural Language Processing
Inference Generation
Text Comprehension
Causal Continuity

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Understanding a story requires the reader to make inferences about things implied by, but not explicitly stated in a story. A reader infers whatever he believes the text implies, but such inferences can turn out to be erroneous. An erroneous inference is one that is contradicted by a subsequent story statement. A reader must be able to correct such erroneous inferences as the story progresses, in order to end up with a consistent and correct understanding of the story. This is especially but not exclusively true of stories with surprise endings, such as mystery stories, fables, jokes.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Next page →

JOB

407 051

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

This thesis is about understanding potentially misleading stories. A reader cannot know ahead of time whether or not a story will turn out to contradict one of its own previous implications. Therefore, virtually every story is potentially misleading. Understanding a story requires the reader to be able to recognize when a story contradicts a previous inference, and to correct the erroneous inference by replacing it with a better inference. ARTHUR (A Reader THat Understands Reflectively) is a computer program that understands stories by inferring unstated connections among the statements in the text, and producing a representation of the story which includes these inferences. The inferences in this story representation must be continually updated in light of each new story statement read. ARTHUR can recognize and correct its own erroneous inferences during understanding, and hence it can understand stories which contain entirely novel information. ARTHUR demonstrates its understanding of a story by using its story representation to answer questions about the story.

ARTHUR embodies a theory of adaptive understanding: it's own understanding processes are affected and altered by what it reads. ARTHUR's ability to understand depends on its knowledge of the situations that can appear in stories, and, reciprocally, its knowledge can be increased according to what ARTHUR reads. ARTHUR's operation is based on a theory of the organization of situational knowledge in an understander's memory, and a theory of the processes by which that knowledge is applied during story understanding.

There are three basic processes underlying ARTHUR's understanding behavior. First, ARTHUR must be able to come up with plausible inferences to connect the statements in a story, while avoiding an exhaustive search through irrelevant information. This is accomplished by using a memory organization in which inference paths are indexed according to their eventual outcomes, thereby allowing appropriate paths to be examined and inappropriate paths to be ignored in any given situation. ARTHUR's first understanding process makes use of this indexed memory organization to efficiently arrive at appropriate plausible inferences.

A plausible inference may turn out to be erroneous. ARTHUR's second basic understanding process is the ability to recognize and resolve contradictions that may arise among its own inferences during understanding.

Finally, understanding a totally novel story may require the use of some previously unknown information, without which the story cannot be understood. AURTHUR's third major understanding process is the ability to integrate new information into its memory, such that that information can be applied in ARTHUR's subsequent inference-making processes.

-- OFFICIAL DISTIRUBTION LIST --

Defense Documentation Center — 12 copies
Cameron Station
Alexandria, Virginia 22314

Office of Naval Research — 2 copies
Information Systems Program
Code 437
Arlington, Virginia 22217

Dr. Judith Daly — 3 copies
Advanced Research Projects Agency
Cybernetics Technology Office
1400 Wilson Boulevard
Arlington, Virginia 22209

Office of Naval Research — 1 copy
Branch Office - Boston
495 Summer Street
Boston, Massachusetts 02210

Office of Naval Research — 1 copy
Branch Office - Chicago
536 South Clark Street
Chicago, Illinois 60615

Office of Naval Research — 1 copy
Branch Office - Pasadena
1030 East Green Street
Pasadena, California 91106

Mr. Steven Wong — 1 copy
New York Area Office
715 Broadway - 5th Floor
New York, New York 10003

Naval Research Laboratory — 6 copies
Technical Information Division
Code 2627
Washington, D.C. 20375

Dr. A.L. Slafkosky — 1 copy
Commandant of the Marine Corps
Code RD-1
Washington, D.C. 20380

Office of Naval Research — 1 copy
Code 455
Arlington, Virginia 22217

Office of Naval Research — 1 copy
Code 458
Arlington, Virginia 22217

Naval Electronics Laboratory Center — 1 copy
Advanced Software Technology Division
Code 5200
San Diego, California 92152

Mr. E.H. Gleissner — 1 copy
Naval Ship Research and Development
Computation and Mathematics Department
Bethesda, Maryland 20084

Captain Grace M. Hopper — 1 copy
NAICOM/MIS Planning Board
Office of the Chief of Naval Operations
Washington, D.C. 20350

Dr. Robert Engelmore — 2 copies
Advanced Research Project Agency
Information Processing Techniques
1400 Wilson Boulevard
Arlington, Virginia 22209

Professor Omar Wing — 1 copy
Columbia University in the City of New York
Department of Electrical Engineering and
Computer Science
New York, New York 10027

Office of Naval Research — 1 copy
Assistant Chief for Technology
Code 200
Arlington, Virginia 22217

Captain Richard L. Martin, USN — 1 copy
Commanding Officer
USS Francis Marion (LPA-249)
FPO New York 09501

Major J.P. Pennell — 1 copy
Headquarters, Marine Corp.
(Attn: Code CCA-40)
Washington, D.C. 20380

Computer Systems Management, Inc. — 5 copies
1300 Wilson Boulevard, Suite 102
Arlington, Virginia 22209

Ms. Robin Dillard — 1 copy
Naval Ocean Systems Center
C2 Information Processing Branch (Code 8242)
271 Catalina Boulevard
San Diego, California 92152

ABSTRACT

ADAPTIVE UNDERSTANDING:
CORRECTING ERRONEOUS INFERENCES

Richard Horace Granger, Jr.
Yale University

Understanding a story requires the reader to make inferences about things implied by, but not explicitly stated in a story. A reader infers whatever he believes the text implies, but such inferences can turn out to be erroneous. An erroneous inference is one that is contradicted by a subsequent story statement. A reader must be able to correct such erroneous inferences as the story progresses, in order to end up with a consistent and correct understanding of the story. This is especially but not exclusively true of stories with surprise endings, such as mystery stories, fables, jokes.

This thesis is about understanding potentially misleading stories. A reader cannot know ahead of time whether or not a story will turn out to contradict one of its own previous implications. Therefore, virtually every story is potentially misleading. Understanding a story requires the reader to be able to recognize when a story contradicts a previous inference, and to correct the erroneous inference by replacing it with a better inference.

ARTHUR (A Reader THat Understands Reflectively) is a computer program that understands stories by inferring unstated connections among the statements in the text, and producing a representation of the story which includes these inferences. The inferences in this story representation must be continually updated in light of each new story statement read. ARTHUR can recognize and correct its own erroneous inferences during understanding, and hence it can understand stories which contain entirely novel information. ARTHUR demonstrates its understanding of a story by using its story representation to answer questions about the story.

ARTHUR embodies a theory of adaptive understanding: it's own understanding processes are affected and altered by what it reads. ARTHUR's ability to understand depends on its knowledge of the situations that can appear in stories, and, reciprocally, its knowledge can be increased according to what ARTHUR reads. ARTHUR's operation is based on a theory of the organization of situational knowledge in an understander's memory, and a theory of the processes by which that knowledge is applied during story understanding.

There are three basic processes underlying ARTHUR's understanding behavior. First, ARTHUR must be able to come up with plausible inferences to connect the statements in a story, while avoiding an exhaustive search through irrelevant information. This is accomplished by using a memory organization in which inference paths are indexed according to their eventual outcomes, thereby allowing appropriate paths to be examined and inappropriate paths to be ignored in any given situation. ARTHUR's first understanding process makes use of this indexed memory organization to efficiently arrive at appropriate plausible inferences.

A plausible inference may turn out to be erroneous. ARTHUR's second basic understanding process is the ability to recognize and resolve contradictions that may arise among its own inferences during understanding.

Finally, understanding a totally novel story may require the use of some previously unknown information, without which the story cannot be understood. ARTHUR's third major understanding process is the ability to integrate new information into its memory, such that that information can be applied in ARTHUR's subsequent inference-making processes.

Preface

In order to understand language, we must first have some knowledge about what we are reading. But in order to have that knowledge, we must have understood it at some previous time. Hence understanding is a "circular" process, in the sense that we must know something before we can understand, but we must be able to understand in order to know things. Understanding a story not only requires a large amount of knowledge about the situations that may occur in the story, but also requires the knowledge that totally unexpected and novel situations may occur in a story. Such novel situations must be both understood and remembered, since they may provide knowledge which is applicable to subsequent understanding. Theories of understanding often presuppose a large body of built-in knowledge on the part of the reader, but those theories rarely if ever specify how that knowledge is affected by the understanding process itself. In particular, when we try to program a computer to understand language, we discover that apparently simple and straightforward processes are in fact immensely complex.

To give a computer the knowledge it needs to understand stories, we need to know what that knowledge looks like. Since stories themselves can contain new knowledge, we need to understand how that information is encoded in language, and how a reader can extract new information from what is read, and how that information can be included in the reader's memory in such a way that it can then be applied to subsequent understanding. Hence, understanding is inherently adaptive: an understander must have a mechanism for extracting new information contained in stories, and a mechanism for then using that information as part of the ongoing understanding process. By equipping a computer program with these abilities, it is possible to build a machine that understands stories which contain entirely novel information.

Acknowledgements

My advisor, Professor Roger Schank, suggested a large number of thesis topics to me, and then he stuck with me when I went my own way. My debt to him is immeasurable, for not giving up on me, and not letting me give up on myself; for giving me the chance to attempt my own achievements, and the chance to make my own mistakes.

Professor Schank is also responsible for compiling the A.I. Project at Yale, which is a tightly knit collection of some of the most high-powered idea-mongers I've ever encountered. I don't know how anyone ever did a dissertation without their help. The A.I. Project is responsible for having brought me into contact with a dazzling array of famous and intriguing people from inside and outside Yale, none of whom ever would have paid the least bit of attention to me had I not been Roger's student.

Professor Drew McDermott and Professor David Barstow waded through some early versions of this thesis. Professor McDermott's first comment was: "I sort of like this stuff ... in spite of the way it's presented." If anyone can make sense out of this thesis, it is primarily due to my long and fruitful dialogues with him. If it is still unintelligible, well, I should have taken up still more of his time.

Thanks to Professor McDermott and Professor Wendy Lehnert for serving on my thesis committee, along with Professor Schank.

The whole thing was an ordeal unlike any I've ever experienced or hope to experience again. Throughout it all, Mallory G. R. Selfridge was more responsible than any other single person for helping me to stay sane. I thank him, and I owe him.

Many other members of the Yale A.I. Project, past and present, have contributed to the ideas in this thesis. They deserve my thanks, one and all: Professor Robert Abelson, Larry Birnbaum, Mark Burstein, Professor Jaime Carbonell, Professor Richard Cullingford, Alan Cypher, Natalie Dehn, Dr. Gerald DeJong, Ann Drinan, Michael Dyer, Margot Flowers, Linda Fusco, Dr. Anatole Gershman, Jim Hendler, Lewis Johnson, Janet Kolodner, Michael Lebowitz, Dr. John McCreery, Rod McGuire, Nat Mishkin, Patti Oronzo, Dr. Chris Riesbeck, Dr. Jerry Samet, Dr. Steve Shwartz, Steve Slade, Professor Rand Spiro, Walter Stutzman, Mike Ward and Professor Robert Wilensky.

There were also many people outside the project who helped me keep going, when the going got weird. They deserve my heartfelt thanks and gratitude. Thanks Cathy, for everything. Thanks to Piglet, Tigger and the rest of the gang ... I won't forget you. Thanks to Jim Meehan and Tim Standish, for giving me a goal to work for above and beyond the thesis itself, and for their patience. Thanks to my sister Patricia, and to my brother Rod, especially for answering the phone at all hours of the night.

And finally, there's my father, the first Dr. Richard H. Granger, and my mother Barbara, who have been my Great Providers through the most trying times of my life. I can't thank them adequately for all of their ideas and encouragement, mental and physical sustenance. I can just say thanks: Thanks, Mom. Thanks, Dad.

"We are what we're trying to study; it's like trying to take a picture of the inside of your camera...."

- David Gerrold,
When HARLIE Was One

"... if I'd a-knowed what a trouble it was to make a book, I never would of done it ...."

- Samuel Clemens,
Huckleberry Finn

TABLE OF CONTENTS

CHAPTER 1

WHY ARTHUR?

1.1 Preface

Stories are often misleading: an implication at the beginning of a story can be contradicted later on in the story. This is especially true of stories with surprise endings, e.g., mystery stories, jokes, fables. Understanding a story requires the reader to make inferences about things that are implied by, but not explicitly stated in a story. Such inferences are only plausible assumptions about a story, based on what has been read so far. Any such assumption may turn out to be wrong, and will then have to be revised as more of the story is read. Hence a story understander must be able to change its mind about its own inferences as the story progresses. Building a computer program that understands stories therefore requires specification of an algorithm for changing one's mind, and the knowledge needed by such an algorithm.

This thesis is about understanding potentially misleading stories. A reader cannot know ahead of time whether a story will turn out to be misleading or not. Therefore, virtually every story is a potentially misleading story. Hence understanding a story requires the reader to first of all be able to determine if and when the story contradicts any previous inferences. This requires the reader to have knowledge about what inferences it has made, what alternative inferences could have been made instead, and a way of measuring which of those alternatives could best account for a new story input. Most of this thesis is concerned with describing the nature of this knowledge, and presenting algorithms that use this knowledge to continually update a reader's interpretation of the story on the basis of what has been read so far.

These algorithms were used to build a story understanding program called ARTHUR (A Reader THat Understands Reflectively). ARTHUR reads stories in English, and produces representations of the stories that include inferences about information which was not explicitly stated

in the story. ARTHUR demonstrates its understanding of a story by using this representation to answer questions about the story.

This chapter introduces the problems involved in understanding misleading stories. Several previous theories of story understanding are examined to see how they deal with these problems. It is concluded that the knowledge needed to understand reflectively is lacking in those previous theories. A brief description of this knowledge is presented, and some problems that arise in its application are discussed.

1.2 Introduction

Understanding a natural language text requires the reader to make inferences to fill in events and information not explicitly stated in the story. For example, consider the following story:

> [1] Ralph Hammond went into the Chesapeake Restaurant. He asked to see the manager, George Jameson. George immediately recognized Ralph as his old high school buddy. Ralph told George that he was in town on business, and he could only stay for a while. They talked and reminisced up until closing, and then Ralph said he had to go. As he was about to leave, Ralph suddenly turned and pulled out a gun. "Sorry, old buddy," he said, and he shot George six times. He ran out to his car, and drove to the pre-arranged payment location. Counting up his money later on, he thought back on the evening's events. He felt depressed for a while, but it didn't last. Being a professional hit man, he didn't have much time to waste on things like feelings.

Q) Why did Ralph go into the restaurant?
A) To find George and shoot him.

To answer question Q, a reader must infer that Ralph went to the restaurant with the plan of shooting George. This connection is not explicitly stated in the story, but human understanders seem to have no trouble inferring it from the events in the story. However, most existing natural language understanding systems would be incapable of reading story [1] and correctly answering question Q. They would not understand the story because they are incapable of

finding the correct inferences to account for the events in the story.

Consider what is involved in arriving at the inference that Ralph's reason for going to the restaurant was to shoot George. First of all, the reader probably made other inferences about Ralph's reasons for going, before making this final inference. To see this, consider what would happen if we asked the reader the same question Q at different points in the story:

[1] Ralph Hammond went into the Chesapeake Restaurant.

Q1) Why did Ralph go into the restaurant?
A1) Perhaps to eat there.

He asked to see the manager, George Jameson. George immediately recognized Ralph as his old high school buddy. Ralph told George that he was in town on business, and he could only stay for a while.

Q1) Why did Ralph go into the restaurant?
A2) To talk to his friend George.

They talked and reminisced up until closing, and then Ralph said he had to go. As he was about to leave, Ralph suddenly turned and pulled out a gun. "Sorry, old buddy," he said, and he shot George six times. He ran out to his car, and drove to the pre-arranged payment location. Counting up his money later on, he thought back on the evening's events. He felt depressed for a while, but it didn't last. Being a professional hit man, he didn't have much time to waste on things like feelings.

Q1) Why did Ralph go into the restaurant?
A3) To find George and shoot him.

In order to answer question Q1 at each point in the story, the reader had to have made inferences about Ralph's reason for going to the restaurant. However, the answers A1, A2 and A3 are all different answers to the same question, based on what events the reader had read so far. Answer A1, for example, reflects simply a default inference which we infer in the absence of any other information. Answer A2 involves the knowledge that people may want to see people who they were friends with at one time but who they haven't seen for a while. Finally, answer A3 is based on the story saying that Ralph killed George and got paid for

it, along with our knowledge that someone who gets paid for some action probably performed the action intentionally.

These three answers to question Q1 are different because the story implies different reasons for Ralph's action at each of these three points in the story. Generating the above answers to question Q1 requires the reader to make each of the implied inferences about Ralph's intentions in turn, with each new inference replacing its predecessor.

1.2.1 The scope of the problem

Understanding story [1] required the reader to generate a series of inferences about a character's intentions, with each new inference replacing the previous one in the reader's overall interpretation of the story. If [1] is representative of a significant number of stories, then it means that our theories of understanding must take into account readers' ability to change their minds about their own inferences in light of subsequent input. Before I elaborate on the elements of this ability and its implementation, it is important to show what types of stories require this ability, and how common such stories are. This section outlines some major categories of stories which contradict their own initial implications. In the process of presenting these classes of stories, the scope of the problem of correcting erroneous inferences will become clearer.

Consider the following:

> [2] John went into the movie theater. He went up to the balcony, where Fred was waiting with an ounce of cocaine. John paid him in hundred-dollar bills and left quickly.

Understanding this example requires the reader to realize that John went into the movie theater for the purpose of consummating a drug deal. This inference is not ordinarily among the most likely possible reasons for entering a movie theater. After the first sentence, the reader's answer to a question about John's intention would reflect the default inference that John probably wanted to see a movie. Like story [1], this story implies a character's intentions, and then subsequently implies a contradictory intention. Understanding this story requires the reader to replace the initial inference with the new one.

This story, and story [1], can be loosely categorized as "mystery stories": they intentionally hide certain information so as to lead the reader to make inferences that

later turn out to be erroneous. Mystery stories constitute one example of the class of stories that are intentionally misleading. That is, stories which are written with the intent of leading the reader down a "garden path", and then suddenly causing the reader to change his mind about the interpretation of the story.

Intentionally misleading stories are very common. Following is a list of some common subcategories of intentionally misleading stories:

(1) A story which surprises a reader for his enjoyment, such as an O. Henry story;

(2) A story which dramatically makes a logical or moral point by reaching a conclusion via a non-intuitive route, such as an Aesop's fable;

(3) A story that startles or frightens a reader, such as a horror story;

(4) A story that sets out a logical puzzle on the basis of a sparse set of clues, such as a detective or mystery story;

(5) A story that amuses a reader, such as a joke.

For example, consider the following version of one of Aesop's fables:

[3] The crow was sitting in her tree holding a piece of cheese, when along came the fox. "Sing for me, Crow," implored the fox, "for you have the most lovely voice in all the world." The crow was much impressed by this flattery. She opened her mouth to sing. When she did so, the cheese fell out of her mouth, and the fox ran away with it, laughing as he ran. The cheese was what he wanted all along.

When we first read the fox's request that the crow sing, we can only infer that the request is genuine, and that he wants to hear her singing. We may suspect that the fox has some hidden plan in mind, but we certainly cannot tell what that plan is from the story at that point. When we eventually see that the cheese falls and the fox grabs it, we can then recognize that the fox's actual goal was to get the crow to drop the cheese. Had this goal been stated at the beginning of the story, we would have made different inferences from the fox's actions.

Consider the following version of the old Henny Youngman joke:

[4] "I took my wife to the Bahamas, but unfortunately she found her way back home."

When we hear the first part of this sentence we infer that the speaker took his wife with him on a vacation. However, the second half of the example says that his wife found her way back home, and implies that he didn't want her to do so. Upon hearing this, we abandon our initial inference, and infer instead that the speaker took his wife to the Bahamas so that she would stay there and be out of his proximity from then on.

If the example instead said "I took my wife to the Bahamas, and we had a wonderful vacation", the natural inference from the first part of the example is substantiated by the second part of the sentence, and no surprise occurs.

In both examples [3] and [4], the story was written in such a way as to lead the understander to first make a "garden path" inference, and then to recognize that that inference was wrong, and finally to replace the inference with a new inference based on new information in the story. Given that a story may be written with the intention of leading the reader down a garden path, we can then ask what types of inferences can turn out to be spurious in such a story.

Even when a story is not intentionally meant to be misleading, a reader may have to make an inference and then change it in light of subsequent input. This can occur when a non-standard activity is performed in an otherwise stereotypical setting. For example, consider the following:

[5] Kathy and Chris were playing golf. Kathy hit a shot into the rough. She wanted to let her good friend Chris win.

Q1) Why did Kathy hit a shot into the rough?
A1) She wanted to let Chris win, so she intentionally put herself at a disadvantage.

Answer A1 requires the reader to infer that Kathy's action was intentionally performed as part of a plan to help her opponent win the game. This inference is not the initial inference that a reader makes from the action of hitting a ball into the rough, as illustrated by the following example:

[6] Kathy and Chris were playing golf. Kathy hit a shot into the rough.

Q1) Why did Kathy hit a shot into the rough?
A2) It wasn't on purpose, she just hit a bad shot.

Example [6] is identical to the first two sentences of [5]. Having read just these two sentences, the reader answers question Q1 differently than after reading the final sentence of [5]. Hence giving answer A1 indicates that the reader must have replaced the initial "not on purpose" inference by the "intentional disadvantage" inference after reading the final sentence of [5].

Previous theories of text understanding, e.g., Charniak [1972], Schank [1973], Schank et al. [1975], Cullingford [1977], Wilensky [1978], have assumed that people make inferences while reading. Furthermore, all of these theories acknowledge that an inference made while reading might turn out to be wrong. However, no previous theory of text understanding has included a method of either recognizing when a previous inference is wrong nor a method for replacing one inference by a subsequent one based on new information in a story.

1.3 ARTHUR

ARTHUR (A Reader THat Understands Reflectively) is a program that can understand misleading stories by re-evaluating its own inferences in light of subsequent input, and correcting inferences which are found to be erroneous. The program understands stories by exercising an ability to change its mind about the inferences it makes in the course of understanding a story. ARTHUR illustrates its understanding by answering "why"-type questions about the stories it has read. Following is some sample input and output from the ARTHUR program:

****

INPUT STORY:
MARY PICKED UP A MAGAZINE. SHE SWATTED A FLY.

INPUT QUESTION:
WHY DID MARY PICK UP A MAGAZINE?

OUTPUT ANSWER:
AT FIRST I THOUGHT IT WAS BECAUSE SHE WANTED TO READ IT, BUT ACTUALLY IT'S BECAUSE SHE WANTED TO USE IT TO GET RID OF A FLY.

****

INPUT STORY:
A MAN CAME TO JOHN'S DOOR. THE MAN SAID HE WANTED TO TALK TO JOHN ABOUT INSURANCE POLICIES. JOHN LET THE MAN IN. THEN THE MAN PULLED A KNIFE AND ORDERED JOHN TO GIVE HIM ALL HIS MONEY.

INPUT QUESTION:
WHY DID THE MAN COME TO JOHN'S DOOR?

OUTPUT ANSWER:
AT FIRST I THOUGHT IT WAS BECAUSE THE MAN WANTED TO TALK TO JOHN ABOUT INSURANCE POLICIES, BUT ACTUALLY IT'S BECAUSE THE MAN WANTED TO STEAL JOHN'S MONEY.

INPUT QUESTION:
WHY DID JOHN LET THE MAN IN?

OUTPUT ANSWER:
JOHN WANTED TO LISTEN TO THE MAN TALK ABOUT INSURANCE POLICIES.

****

INPUT STORY:
JOHN WAS HUNGRY. HE WENT INTO MCDONALD'S. HE DIDN'T EAT ANYTHING. JOHN ONLY EATS MACROBIOTIC FOOD.

INPUT QUESTION:
WHY DIDN'T JOHN EAT ANYTHING?

OUTPUT ANSWER:
BECAUSE HE WANTED TO EAT MACROBIOTIC FOOD, AND MCDONALD'S DIDN'T HAVE MACROBIOTIC FOOD.

****

In order to build the ARTHUR program, I had to develop a theory of how understanders re-evaluate their own inferences to account for new story input. The theory gives the knowledge an understander needs and the rules for applying this knowledge. The theory is embodied in the implementation of the ARTHUR program, but it is also intended as a possible model of how human understanders might perform these processes.

1.4 Brief overview of other work on inference

Many previous research efforts in natural language processing have involved inference mechanisms for story understanding. The notion of the role of inference in understanding dates back at least to Raphael [1964], and has been enormously elaborated in the subsequent work of Charniak [1972], Winograd [1972], Schank [1973], Rieger [1975], Cullingford [1977], Schank and Abelson [1977] and Wilensky [1978], to name but a few. The work in this thesis is very much in the Conceptual Dependency paradigm of representations of natural language texts. Hence, this thesis draws heavily on much of that previous work, and some knowledge of that work will be very helpful in understanding the work presented here.

1.4.1 Rieger [1975]

Rieger [1975] describes a theory of understanding implemented in the MEMORY module of the MARGIE system (Schank et. al. [1973]) Rieger's theory suggested that a reader could potentially make sixteen inferences from every statement of an event, and that each of those inferences then became input to the inference process again, resulting in 16 X 16 or 256 inferences, each of which again became new input to the process. Each generation of inferences had an associated numerical "strength", which diminished with each iteration. The inference process finally halted when the strength of the inferences fell below some arbitrary level.

Rieger's program generated these inferences in order to find implied connections between statements in a text. The connections were based on Schank's [1973] theory of causal chains as a representation of connected text. There are two basic problems with this approach:

1 - The number of inferences generated to find a simple connection is astronomical. Rieger estimated that 500-1000 inferences would be generated from a simple utterance.

2 - The type of inferences generated are very low-level inferences about the immediate antecedents and consequents of events. No larger inferential frameworks containing general information about story situations or human intentions are discussed in Rieger's work. Some stories presuppose knowledge of typical situations and interactions between people. Rieger's inference classes could not be used to understand stories which required such knowledge.

1.4.2 Cullingford [1978]

Some of the problems with Rieger's work were addressed by the SAM system, described in Cullingford [1978]. SAM had knowledge of the event sequences implied by certain stereotypical situations. These event sequences, called scripts, were used to fill in the unstated events implied in stories involving stereotypical situations such as taking a plane trip or going to a restaurant. For example, consider the following:

> [7] John was going on a business trip to New York. He went to the airport and got a ticket. When the plane landed, he went to his hotel and asked the porter to carry his bags. The next day, he was back home in L.A.

SAM's inference mechanism worked by attempting to match every new input in a story with some known event in the script. Script events that were not mentioned in the text were inferred to have been performed. Hence in the above story SAM would infer that John boarded the plane, that the plane took off, and that the porter carried John's bags to his room. These inferences can only be made by using specific knowledge about plane trips and hotels. This script knowledge is an example of the sort of inference that Rieger's theory does not model. The advantage of the script approach is that few inferences need to be generated from the story statements in order to build a story representation that includes implied but unstated information.

There are two main problems with a script-based understander like SAM:

1 - SAM understands only stories which conform closely to the stereotypical event sequence defined by the script. Hence SAM cannot understand novel stories, i.e., stories which diverge in any significant way from the script.

2 - SAM has no knowledge of *why* the events in the script are taking place. For example, in the above story, SAM does not understand the relationship between John's plane trip and his intention to keep a business appointment. Hence SAM cannot understand complex interactions between story characters, without having pre-defined scripts for every such interaction that might take place.

1.4.3 Wilensky [1978]

Wilensky's [1978] PAM program addressed some of the difficulties with the script-based approach. PAM had some knowledge of human intentionality, based on Schank and Abelson's [1977] theory of the *goals* and *plans* underlying purposeful actions. PAM was much more flexible than SAM, and could understand non-stereotypical situations by inferring the intentions underlying story characters' actions. Consider the following example:

> [8] John had a business meeting in New York. The planes were all grounded because of fog. He decided to hitchhike. When he finally got to his hotel, the porter refused to carry his bags. John slipped him twenty dollars, and the porter became more friendly.

PAM could understand John's actions as being novel methods of achieving his goals. His first goal was to get to New York, and his next was to get his bags moved to his room. John experiences difficulty with achieving these goals by the stereotypical methods, and resorts to other plans to achieve his goals.

PAM works by inferring story characters' goals, and then attempting to find a sequence of inferences from a character's actions which can connect each action to his goals. In this way, the PAM program attempts to *explain* each of the character's actions as having been performed in service of a certain goal.

1.4.4 Problems

One problem with PAM as an understander of novel stories is that it cannot change its mind about its interpretation of a story. Hence it cannot correctly interpret a story which is misleading. Section 1.1.1 illustrated the fact that a number of classes of novel stories are misleading by nature. Since PAM never takes back an inference once it has been generated, it would be unable to understand a story which contradicted one of its own previous implications. For example, consider the following:

[9] John went to the gas station. He robbed it and got away with $50.

A reader could answer a question about John's intentions after the first sentence, and again after the second sentence of [9]. The answers to the question would be different depending on when it was asked. This is because the most likely inference from the first sentence is that John wanted to use the gas station for its typical purpose of supplying gasoline. A reader's answer to a question about John's intentions after the first sentence would reflect that inference. After the second sentence, the reader would answer that John went to the station in order to rob it.

Example [9] poses a number of serious problems for an understander like PAM. PAM is capable of understanding stories in which a story character changes his mind. For example, if John first wanted to get gas, and then decided to rob the station instead, PAM could infer that John may have had two mutually exclusive goals, and he decided which one to act on. Schank and Abelson [1977] describe goal fate graphs, which trace the history of a character's goals as they change in the course of a story. In example [9], it is clearly a misunderstanding of the story to infer that John had the intention of getting gas, and then changed his mind and decided to rob the station instead. Rather, it is we, the readers, who first attribute one goal to John on the basis of our default knowledge about gas stations, and we then change our minds about that inference. A representation of this change constitutes an inference fate graph, corresponding to the inferences we made and discarded in the course of understanding the story. This thesis proposes a method of maintaining and updating a type of inference fate graph, called an explanation graph, during understanding. Explanation graphs are introduced in Chapter 2.

1.5 Changing your mind

What can be done to build a story understanding system that can read and understand potentially misleading stories? In order to see what is necessary to build such a system, it is useful to examine the problems that PAM has with such stories.

The major limitation of the PAM system is that its inference mechanism is based on the assumption that stories will not contradict their own previous implications. The problem is that virtually any story might contradict one of its own earlier implications, and understanding such a story requires the reader to be able to re-evaluate the validity of its own previous inferences. For example, recall the following:

> [2] John went into the movie theater. He went up to the balcony, where Fred was waiting with an ounce of cocaine. John paid him in hundred-dollar bills and left quickly.

To understand [2], the reader must recognize that one might enter a movie theater for almost any purpose. In the absence of any other information, the best default guess is that John wanted to see a movie, since that is known to be the primary purpose of movie theaters. However, other information in a story may contradict that inference. The reader must be able to recognize when such a contradiction occurs, and must be able to update its own inferences to take the new information into account.

The PAM program could not have understood [2] because PAM does not ever consider the possibility that one of its own inferences might have been erroneous. Once PAM has made an inference about a character's probable intention, it never re-evaluates the validity of that inference. To understand story [2], the reader must know that its own previous inferences are only plausible assumptions about John's intentions. Behind each such plausible assumption are myriad other alternate possible inferences that could have been generated, but which were deemed less likely at the time the inference was made.

To account for an apparently contradictory story input, one possibility that the reader must consider is that one of its own previous inferences may have been erroneous. This requires the reader to be able to re-evaluate its own inferences. This re-evaluation ability is one important aspect of adaptive understanding. Adaptive understanding includes being able to make inferences about what is not stated in the story, re-evaluating those inferences in light of subsequent input, replacing an inference to account for an apparent contradiction, and updating the overall

interpretation of the story to be consistent with the new inference.

1.6 When to change your mind/ What to change it to

In story [2] above, the reader's initial inference was that John was going to the movie theater to see a movie. The reader then decided that that inference was erroneous, and that John was actually going to the theater to make a coke deal.

Consider what is involved in this decision. First, the reader must recognize that the initial inference is erroneous, i.e., John did not go into the theater to see a movie. There is no obvious reason why this inference should not still be considered valid. For example, the story might have been as follows:

> [10] John went into the movie theater. He went up to the balcony, where Fred was waiting with an ounce of cocaine. John was surprised to see him. He hadn't known there was any good coke in town at all. He decided to take advantage of the opportunity. John paid Fred in hundred-dollar bills and left quickly.

This story is identical to [2], with three sentences inserted into the middle of the story. These three sentences state explicitly that John had not gone to the theater for the purpose of making the deal. Hence, the initial inference is still perfectly valid: John probably did go to the theater to see a movie.

The point here is that John's action of buying coke from Fred <u>does not explicitly contradict the initial inference about John's intention</u>. It is possible to infer that John showed up at the theater fortuitously, and then decided to buy some coke. This is true of the original story as well: it is possible that John went to the theater to see a movie, then ran into Fred, and decided to buy some coke from him. There is nothing in the story to rule out this possibility. Yet we would all agree that the story seems to imply that John went to the theater for the purpose of meeting Fred there.

Hence the decision to replace a previous inference by a new one is not based on the explicit contradiction of that inference by subsequent information in the story. Rather, the subsequent information implies that another inference might be preferable, and the reader responds to this implication by replacing the old inference with the new one. This results in a story representation containing a single

inferred goal which accounts for both of John's actions, rather than two separate inferred goals each accounting for one of his actions.

The reader prefers this more parsimonious story representation over the alternative less parsimonious one, in spite of the fact that this requires the reader to do the extra work involved in replacing one of its previous inferences. This is not an isolated phenomenon; it is applicable to virtually all misleading stories. Consider the following deceptively simple example:

[11] Mary picked up a magazine. She swatted a fly.

If asked why Mary picked up the magazine, a reader would answer after the first sentence that Mary probably wanted to read the magazine, but after the second sentence, the answer would be that she wanted to swat a fly with the magazine. There is nothing in the story to make the reader prefer this interpretation over the interpretation that Mary may have picked up the magazine to read it, and then she was bothered by a fly so she swatted it with the magazine. In fact, the story doesn't even say that she used the magazine to swat the fly. Hence, the following three interpretations are all valid on the basis of what the story says:

(11a) Mary picked up a magazine to read it. She then was annoyed by a fly, and she swatted it with the magazine she was holding.

(11b) Mary picked up a magazine to read it. She then was annoyed by a fly, and she swatted it with a flyswatter that was handy.

(11c) Mary picked up a magazine to swat a fly with it.

The last interpretation (11c) reflects a story representation which consists of a single goal, getting rid of a fly, which both of Mary's actions were performed in service of. The other interpretations both consist of two separate goals, each of which explains one of Mary's actions.

In [11], as in [2], the interpretation generated by the reader is the most <u>parsimonious</u> of the possible interpretations. That is, the preferred interpretation is the one in which the fewest number of inferred goals of a story character account for the maximum number of his actions. We summarize this observation in the following principle:

The Parsimony Principle

> The best representation of a story is the one requiring the fewest number of goals to explain the actions of a story character.

1.7 Maintaining a parsimonious explanation

ARTHUR can understand stories which require it to change its mind about inferences as it reads. In order to build ARTHUR, I had to develop a theory of how the information in a story could be represented in order to allow the program to understand stories which required it to change its mind about its own inferences.

We will use the term explanation to mean the completed representation of a story which contains information implied by but not explicitly stated in the story. An intuitive notion of an explanation for a story statement involves "accounting" for the presence of each statement, by relating it to other parts of the story, both stated and implied. ARTHUR recognizes a subset of this general notion, in which every event must be explained by its connection to a stated or inferred goal in the story, via an inference path consisting of plans, scripts or causal state changes.

Since understanding a story correctly requires finding the most parsimonious explanation, an understander must have a metric for determining when an explanation is parsimonious and when it isn't. In ARTHUR's taxonomy of explanations, a parsimonious explanation is one in which the fewest goals are needed to explain all of the events in a story representation. Maintaining a parsimonious explanation therefore requires ARTHUR to continually update its representation of a story in such a way as to parsimoniously incorporate each new story statement into the representation.

There are many different types of statements that can occur in a story: for example, events, goals, states, causal chains. Depending on the type of input statement, ARTHUR will react differently. Different types of input require different types of inferences to be made to account for the input satisfactorily.

ARTHUR recognizes three broad categories of story statements:

1 - actions and states
2 - goals
3 - inference rules

In this section, we will outline the scope of the story statements that can be described by these three categories, and we will examine the behavior of human understanders and ARTHUR in reaction to statements in each of these three categories.

1.7.1 The story input is an action or a state

Consider the following:

[12] John picked up a cigarette.

[13] John wanted to give a cigarette to his friend Jim. John picked up a cigarette.

Once we have read the statement of John's intention in [13], we then interpret his action of picking up a cigarette differently than in the absence of that statement, as in [12]. Example [13] leads us to infer that John picked up the cigarette in order to give it to Jim, but [12] leads us to infer that John may have wanted to smoke the cigarette himself.

In general, when both a goal and an action of a character have been stated in a story, then an understander attempts to interpret the action in terms of the goal: that is, the action may have been in service of the goal. Arriving at such an interpretation involves determining whether there exists a sequence of inferences which can connect the action and the goal. Since there are many such inferences from any given action and goal, interpreting an action in terms of a goal involves specification of the correct inference path from among the myriad alternative paths.

Specification of a connective inference path is essentially a search procedure through the space of possible inferences from a goal and an action. The problem with specification is to avoid the potential combinatorial explosion that can result from searching such a large space.

To give a feel for what statements are considered by ARTHUR to be events or states, the following is a list of some examples of story inputs that are in this class:

"Mary picked up a magazine."
"John sold his watch at the pawn shop."
"Sam was at the store."
"Joanie went to Boston."
"Bob ate a hamburger."
"Roses are red."
"I sat down."

1.7.2 The story input is a goal

Consider the following:

[14] John wanted to win the backgammon championship.

This is a statement of John's goal, with no mention of any actions on his part. We know, however, that this goal may be used to interpret subsequent actions that John may perform. In anticipation of this use of the goal, we can make predictions from the goal as to what might happen next in the story. These predictions aid us in interpreting subsequent actions. We might predict that John will perform some actions to aid him in the championship. Hence if the next sentence in this example said that John decided to weight some dice, we would be able to infer that he was intending to cheat in a game of backgammon, in order to win the championship.

The following is a list of some story inputs that are statements of goals, explicitly or implicitly:

"John wanted to own a Mercedes."
"Willa was hungry."
"Bill wanted to win the stockcar race."
"Sandy was horny."
"Jack was bored."
"I was tired."

Predictions from a goal are not absolute assertions about what will happen in the story. For example, if John weights some dice it may be that he is intending to plant them on his opponent, to discredit him, as opposed to using them for his own direct benefit. This is simply another way of saying what we have been saying all along: that subsequent input in a story may force us to change our minds about the validity of a previous inference.

Changing our minds about a previous inference involves supplanting that inference by a new replacement inference. The problems involved in supplanting an inference are threefold; the understander must be able to:

(1) recognize that a previous inference was erroneous
(2) find a corrective replacement for that inference
(3) update the story representation to reflect the correction

1.7.3 The story input is a new rule of inference

Consider the following:

[15] John eats ants.

This story contains a previously unknown fact about the world: that a particular actor, John, eats an object which is not known to be a food, ants. This fact contains information about an object (ants), and about an actor's plan (eating) in service of a goal (satisfying hunger).

Facts about objects and about ways of satisfying goals are encoded in ARTHUR as inference rules: ARTHUR generates inferences from what it reads based facts such as these. For example, the fact that a gun can be used to threaten people is encoded as an inference rule, which can be expressed as follows:

If a character gets control of a gun
then infer that he may intend to threaten someone with the gun.

Hence, new facts in a story can be encoded in ARTHUR in the form of new inference rules. For example, once the reader knows the above fact about John and ants, then whenever the story next says that John got some ants, the reader can infer that John may plan to eat the ants. This inference is not one that would have been made prior to the introduction of this new fact, because the reader had no inference rule which would generate eating as a bottom-up inference from the action of getting ants. Hence making this inference requires the reader to add a new inference rule to its own memory, such that that rule will give rise to this new inference.

Following is a list of some more examples of story inputs that are interpreted as inference rules:

"Drinking hot chocolate makes John feel sleepy."
"Grey mushrooms are poisonous."

"Billy always hurts himself when he plays Ultimate Frisbee."
"Pomegranates are Jenny's favorite fruit."
"Wearing bell-bottomed pants helps keep you warm in cold climates."
"Most of John's in-laws are crooks."

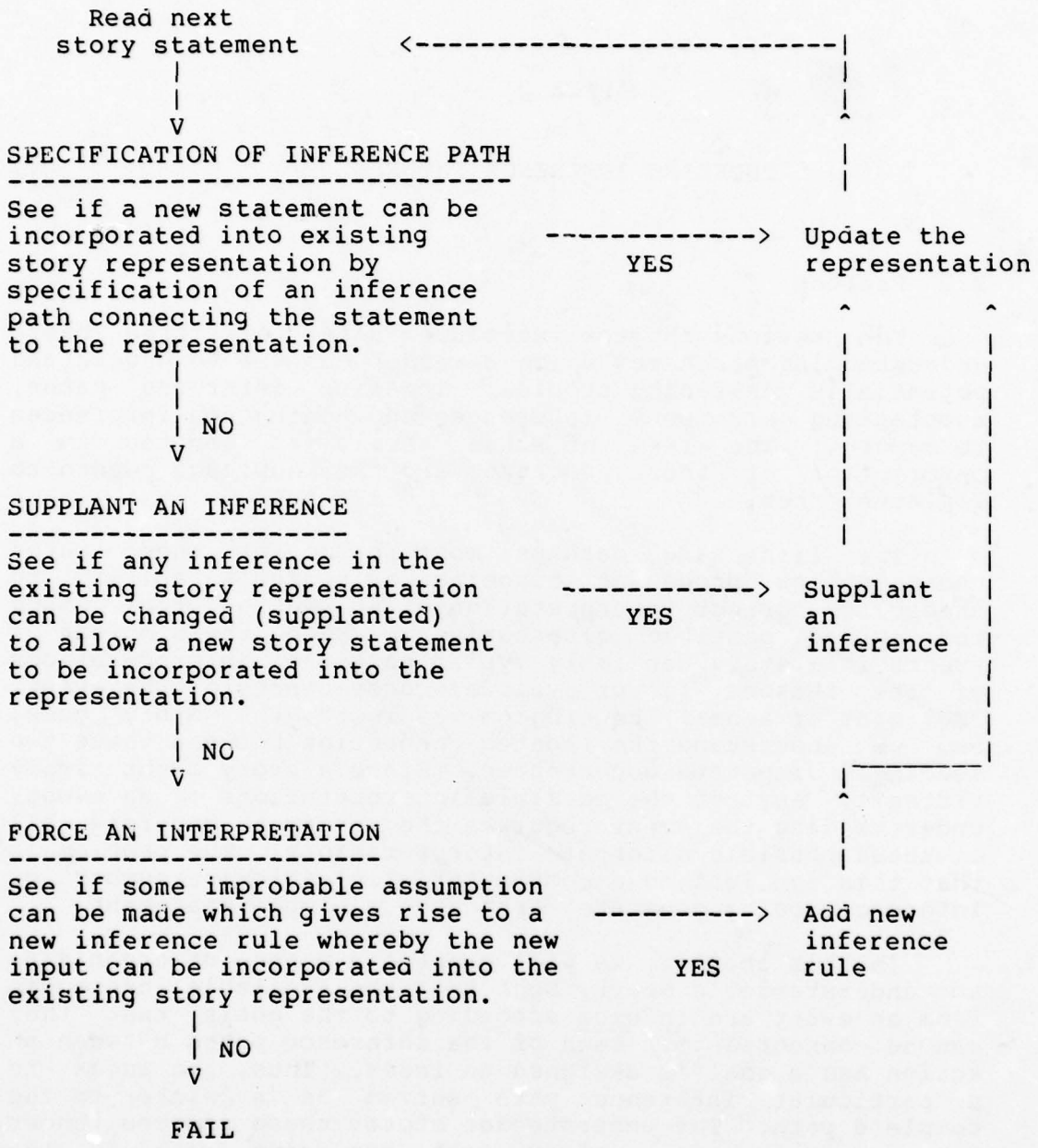
1.8 Subprocesses of the understanding process

We have seen the rules by which an understander reacts to the three types of story input: events, goals and inference rules. I have identified three major subprocesses which comprise the understanding behavior outlined in the previous sections. These processes are as follows:

1 - Specification of an inference path
2 - Supplanting an inference
3 - Forcing an interpretation

The processing of each new statement in a story may call on any or all of these subprocesses. The following flowchart of ARTHUR's processing algorithm outlines the use of these three processes during understanding. The rest of this thesis consists of a detailed investigation of these three subprocesses of the understanding process.

ARTHUR's processing algorithm



[Figure 1]

CHAPTER 2

INDEXING INFERENCE PATHS

2.1 Preface

The previous chapter introduced a set of three basic understanding processes which a reader can use to understand potentially misleading stories: indexing inference paths, supplanting erroneous inferences and adding new inferences to memory. The rest of this thesis is devoted to a description of these processes and the knowledge needed to implement them.

The first and perhaps most basic of these three understanding processes concerns a reader's ability to choose the correct interpretation of an action from among the myriad possible alternatives. Even the simplest of events in a story can imply myriad possible interpretations of the reasons for or eventual consequences of the event. "For want of a nail the kingdom was lost", the story goes, and we understand the imputed connection between these two seemingly disparate occurrences. Since a story might imply virtually any of the possible interpretations of an event, understanding the event requires the reader to generate all of these possible alternate interpretations. The problem is that this can lead to a combinatorially explosive number of inferences being generated from even a simple statement.

In this chapter, we will present a method of organizing an understander's memory such that the available inferences from an event are <u>indexed</u> according to the goals that they can be connected to. Each of the inference paths between an action and a goal is assigned an index. Thus, the index to a particular inference path serves as a pointer to the complete path. The understander stores these indices under the concepts in memory that can give rise to the corresponding inference paths. These indices then comprise both the bottom-up inferences from actions and the top-down inferences from goals. By comparing these indices against each other, the understander can tell whether or not an inference path exists between an action and a goal. This comparison process is called <u>specification search</u>.

Indexing inferences allows ARTHUR to keep a trace of its own understanding processes. The representation of this trace is in an explanation graph. We will see in Chapter 4 that the understander's access to this explanation graph is a crucial part of the ability to "change its mind" about an inference it had previously made in a story.

2.2 Introduction

Consider the following stories:

[16] "I'm so bored," thought Mary.
She picked up the New Yorker magazine.

[17] "I need to jot down John's telephone number." thought Mary.
She picked up the New Yorker magazine.

[18] "This room is full of filthy flies," thought Mary.
She picked up the New Yorker magazine.

[19] "This room is a mess," thought Mary. "Time to clean it."
She picked up the New Yorker magazine.

A reader will infer a different reason for Mary's action of picking up the magazine in each of these examples. For instance, if asked why Mary picked up a magazine after reading example [16], the reader would answer that she probably wanted to read it. After [17], the answer is that she probably wanted to use it to write John's number on. Based on [18], she probably wanted to swat a fly with it, and from [19], she most likely planned to throw it away as trash.

Hence, the correct inference as to why Mary picked up a magazine depends on the previously implied goal. In order to correctly interpret a character's action, the reader must be able to choose from among the myriad possible reasons why that action could be performed. This requires the reader to be able to generate all of the possible alternative inferences, since it cannot be known ahead of time which of these alternatives might prove to be the correct one in any particular instance. Furthermore, the reader must be able to choose the correct interpretation from among the alternatives, based on previous inferences.

There is only a finite amount of information conveyed by Mary's action of picking up a magazine. Specifically, there is an action, which we represent as the Conceptual Dependency GRASP, an actor, Mary, and an object, the

magazine. Any inferences the reader makes from the action must be derived from some combination of these three elements. These bottom-up inferences from the action can then be interpreted in many different ways when they are considered in combination with the top-down inferences from the previously inferred goals of the actor. The problem is to determine what information a reader uses to arrive at a particular connection between the action of grasping the magazine and the particular goal (e.g. cleaning up a mess), from among the many possible combinations of top-down and bottom-up inferences that can be generated.

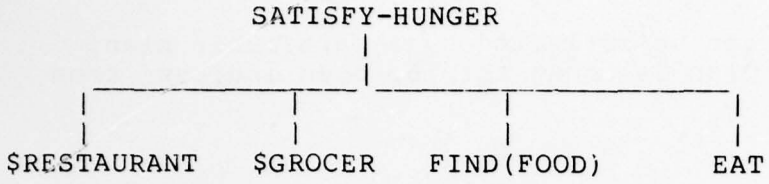
2.3 Indexed inference paths

Consider story [20]:

[20] Willa was hungry. She picked up the New Yorker magazine.

Having read this story, an understander could infer that Willa may have picked up the New Yorker in order to find a place to go out to eat. This explanation of course assumes that the understander knows that information about restaurants is in the New Yorker, but we also must realistically assume that the understander knows that the New Yorker contains information about movies, shows, and nightclub acts, as well as containing news stories, commentary, fiction and poetry, none of which is relevant to satisfying one's hunger. Furthermore, the understander presumably also knows that a New Yorker magazine can be used for functions other than reading; for example, it could be tacked up on the wall, or used to swat a fly, or re-sold for a profit. The problem is, how can the understander recognize the connection between wanting to eat and looking in the New Yorker, without having exhaustively predicted all these possibilities?

The act of picking up any functional object implies the possibility that the actor will use some function of that object. This rule of understanding was suggested by Wilensky [1978]. The problem we are faced with here is that we don't know what function of the New Yorker Willa is going to use. Yet the information that she was hungry should help us to pick the right function. How can this be done? Consider some of the possible plans that can be performed in service of a SATISFY-HUNGER goal:



Here are four different (though not mutually exclusive) paths that an understander can follow from SATISFY-HUNGER: doing the $RESTAURANT script, the $GROCER script, the FIND(FOOD) plan or the EAT plan. (EAT is a plan described in Wilensky [1978]. It consists of the following short sequence of actions: GRASP food, PTRANS food TO mouth, INGEST food.)

These possible paths are all built into an understander; i.e., they are already known in their entirety before a story is read. This is true of PAM and ARTHUR, and, we hypothesize, it is true of human understanders as well. In ARTHUR we take advantage of this fact by explicitly storing this knowledge under the memory entry for SATISFY-HUNGER. In other words, the four scripts and plans listed above, $RESTAURANT, $GROCER, EAT and FIND(FOOD), are stored directly under the SATISFY-HUNGER entry in ARTHUR's memory. The only difference between this implementation and that of the PAM program is that PAM must more circuitously re-generate these four plans from SATISFY-HUNGER, while ARTHUR pre-stores them and therefore can generate them by looking them up in memory directly.

For every goal in its memory, ARTHUR has a set of TOP-DOWN INDICES which aid in directing the processing of stories. These indices can be any memory structure which can connect a goal to an action. According to the syntax of explanations offered by Wilensky [1978] and Schank and Abelson [1977], the only memory structures which serve to connect an action to a goal are scripts and plans. Hence the top-down indices from a goal consist of the set of scripts and plans that connect this goal to various actions. This is summarized in the following two rules:

Rule 1: Top-down script indices

If a goal can be satisfied by a particular script,
Then that script is among the top-down indices
from the goal.

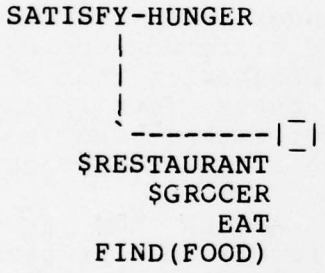
Rule 2: Top-down plan indices

If a goal can be satisfied by a particular plan,
Then that plan is among the top-down indices from the goal.

The top-down indices under SATISFY-HUNGER are as follows:

```
SATISFY-HUNGER: TOP-DOWN INDICES
--------------------------------
$RESTAURANT
$GROCER
EAT
FIND(FOOD)
```

Whenever a goal is stated or inferred in a story, ARTHUR generates a representation of both the goal and this set of top-down indices, as predictive inferences. We call such representations explanation-graphs. Following is the explanation graph for the goal SATISFY-HUNGER:

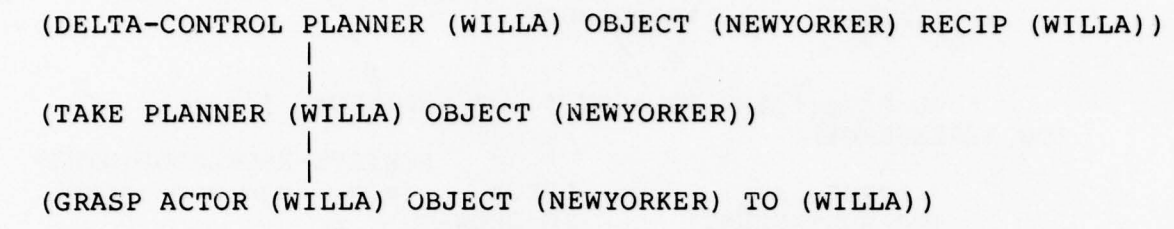


[Figure 2]

These four top-down indices point to the possible plans that could be predicted for an actor with a SATISFY-HUNGER goal. The first sentence of [20], "Mary was hungry", implicitly states the goal of SATISFY-HUNGER. ARTHUR predicts all four of these plans after reading that sentence and generating that goal. We then go on to the second sentence: "She picked up the New Yorker".

The action in this sentence is GRASPing the New Yorker. In a similar example in Wilensky [1978], PAM attempted to infer bottom-up from this action, inferring the possible plans and goals that the action might have been in service of. In this case, a GRASP is inferred to be in service of a DELTA-CONTROL goal, which in turn is in service of a READ

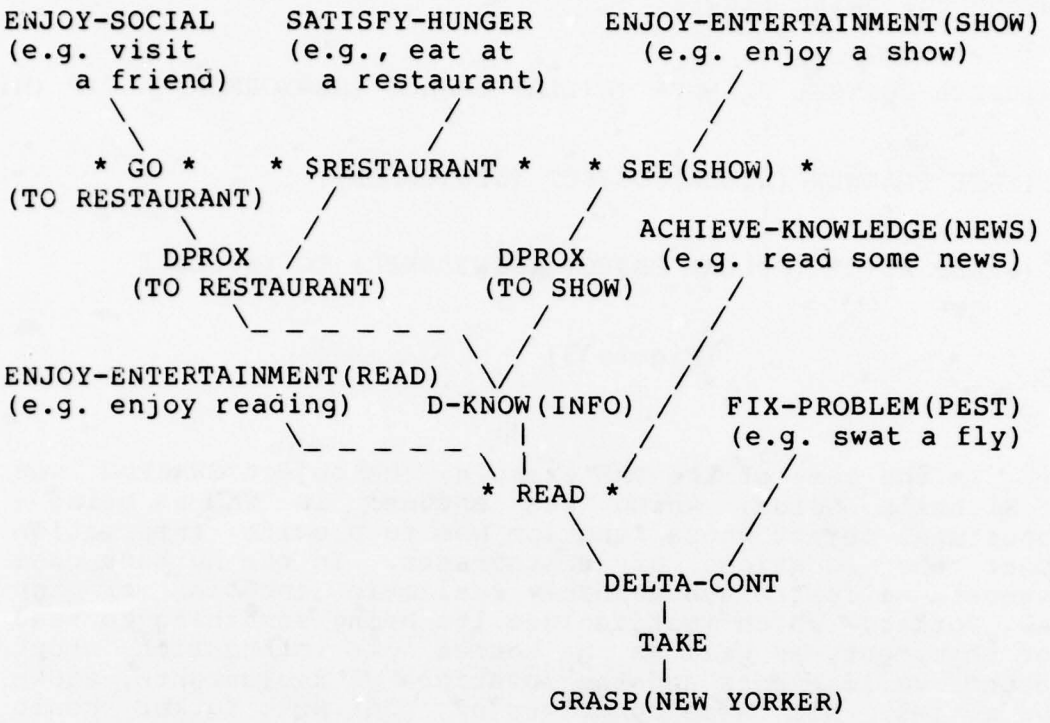
plan, which is for the purpose of DELTA-KNOWing some information. The inferences thus far generated are depicted in the following figure:



[Figure 3]

In the case of the PAM example, the object GRASPed was a Michelin Guide, which was encoded in PAM as being a functional object whose function was to provide information about the locations of restaurants. In the current case however, we desire a reasonably realistic encoding of the New Yorker, which must include its being something to read for enjoyment, as well as a source of information about recent world events and the locations of restaurants, shows and movies. All these functions of the New Yorker could lead to different possible bottom-up lines of inference once the understander has gotten past the DELTA-CONTROL inference.

PAM handled all this information by generating all alternative bottom-up inference lines and keeping track of all the possible lines in parallel. When one of those inference lines resulted in a connection with an existing goal, namely the one about restaurants, then all of the other parallel lines would be discontinued, and a single satisfactory explanation would have been found. However, it would be nice if at the time the alternative possibilities were noticed, the understander had some information which might pare down those possibilities. Just as we have postulated top-down indices for goals, which contained information about what plans could be in service of those goals, we postulate similarly indexed information for actions, namely, what plans they may be in service of. This information is encoded in BOTTOM-UP INDICES from actions. In this case, the action is GRASP, and we can reliably infer that the GRASP is in service of a TAKE plan, which is for a DELTA-CONTROL goal as shown above in Figure 3. However, then there are a number of possible reasons why Willa mignt have wanted to DELTA-CONTROL the magazine. The following figure gives the possible bottom-up inferences from the GRASP action:



[Figure 4]

(We adopt the convention (from Wilensky [1978]) of inferring a "TAKE" plan between the GRASP action and the DELTA-CONTROL goal. This plan actually adds no new information, but it keeps the "syntax of explanation" straight: action - plan - goal - plan - goal etc.)

We can see that the first two bottom-up inferences from the GRASP involve no choices and therefore they also add no new information. Then a tree of possibilities develops from the DELTA-CONTROL inference. We want to index each of these possible paths by plans or scripts, such that each index denotes a different possible explanation for this action. The penultimate plan in each bottom-up explanation sequence, i.e., the plan immediately before the final goal explanation, provides such an index. These plan indices are marked by a star (*) in the figure. In this case those are $RESTAURANT, SEE(SHOW) and READ. (SEE(SHOW) consists of mutual ATRANSing a ticket at a box-office, PTRANSing to a seat, and ATTENDing eyes to the movie or show. READ denotes GRASP book or other readable material, ATTEND eyes and MTRANS from the book to the actor. The reason READ is a

separate index is that it can lead to an explanation of ENJOY-ENTERTAINMENT all by itself, without passing through any other subsequent plan.

Rule 3: Bottom-up indices

If an action can be part of a particular plan or script,
Then that plan or script is among the bottom-up indices from the action.

Of course, an understander can never hope to pre-store all of the possible inferences that could be made from a given action. To illustrate this, consider some of the possible actions that can be performed with a magazine. The following is only a small subset of the total set of possible actions:

1 - To read
2 - To write a note to yourself in a margin
3 - To swat a fly
4 - To line the garbage can
5 - To decorate your wall
6 - To prop up a leg of a table
7 - To wipe up a spill
8 - To throw at someone

There is no point in continuing to add possible uses to this list; the point is that there are a large number of non-standard uses of magazines. Understanders are capable of understanding stories which contain non-standard uses of objects. A later section of this chapter will introduce the notion of precondition indices, which are used to understand non-standard uses of objects.

The basic idea underlying precondition indices is that an actor can plan to have any object, functional or non-functional. If an actor gets control of an object, but then the actor doesn't use the standard function of that object (if any), then the action of getting the object can still be interpreted as being part of any plan in which that object plays any role whatsoever. For example, consider the following:

[21] John got a dictionary. He stood on it so he could reach the top shelf.

The normal function of a dictionary is to find out some information. Standing on the dictionary is not a standard, pre-compiled known function of a dictionary. However, since the dictionary appears in the second action, as the object stood on, the first action of getting the dictionary can be

inferred to satisfy a DCONT PRECONDITION for the action of standing on it. Simply by virtue of an object's being used in any way, that use can be recognized as a valid reason for having gotten control of the object. Similarly, getting to a location or finding out some particular piece of information can be interpreted in a non-standard way by using precondition indices. More will be said about precondition indices in a later section, and in Chapter 3.

2.4 Looking up indices in memory

We have seen that the bottom-up indices from picking up a New Yorker magazine are READ, SEE(SHOW) and $RESTAURANT. We do not want the understander to store these indices under the GRASP action directly; the indices are much too specific to arise from the much more general GRASP action. The action of GRASPing might give rise to any number of bottom-up inferences, depending on other factors such as what was grasped and who did the grasping.

In fact the bottom-up indices given here depend directly on the OBJECT slot of the GRASP, i.e., the magazine. Hence the indices under GRASP must exist in the form of rules which tell the understander where to look up the appropriate indices in memory. These rules will specify which of the rolefillers in the instantiated GRASP conceptualization should be accessed to arrive at the correct bottom-up index in a given instantiation. Hence the bottom-up indices for GRASP are generated by the following rules:

Rule 4: Controlling a functional object

If an actor performs an action which results in his having control over a functional object,

Then infer that the action may be part of a plan to perform the function of the object.

Rule 5: Inferring a known plan

If a character has a known plan,
and that character performs an action,

Then infer that the action may be part of the plan.

ARTHUR has some pre-defined attributes of certain known objects and actors stored in its memory. Implementing the above rules therefore consists of looking up these attributes in memory, and generating whatever inference is

stored there. In particular, ARTHUR stores the functions of functional objects under the FUNCTION attribute of the memory entry for the object; and it stores the known plans of characters under the PLANNER attribute of the memory entry for that character. Since the above rules are implemented in terms of simply looking up these attributes of memory entries, the rules can be expressed in the following way:

GRASP: BOTTOM-UP INDICES

Rule 4a: Functional objects
Infer indices in FUNCTION attribute of OBJECT slot of GRASP

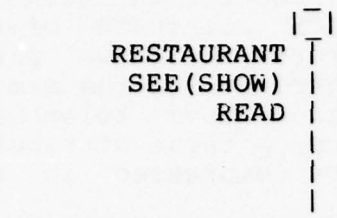
Rule 5a: Known plans
Infer indices in PLANNER attribute of ACTOR slot of GRASP

In this case, the OBJECT slot of the GRASP is the New Yorker, which we said had information about restaurants, movies and shows as well as fiction entertainment. That information is encoded statically under the entry in memory for the New Yorker magazine. The memory entry for "New Yorker" already exists, and contains precisely the functional information needed by the above rule for GRASP. All the rule does is to look up that information under the memory entry for New Yorker. Hence the application of the above INDEX RULE will access the list of pre-computed FUNCTIONs of the New Yorker magazine in memory, resulting in the following indices for this GRASP conceptualization in context:

(GRASP ACTOR (WILLA) OBJECT (NEW YORKER)):
EXPANDED BOTTOM-UP INDICES

$RESTAURANT
SEE(SHOW)
READ

Just as we represented the top-down predictive inferences from goals in explanation-graphs, there is a corresponding explanation-graph representation for bottom-up inferences from actions. The above bottom-up index for GRASP is represented in an explanation-graph as follows:



(GRASP ACTOR (WILLA) OBJECT (NEW YORKER))

[Figure 5]

Of course, many other actions could be performed with the New Yorker as the OBJECT of the GRASP action. For example, we could swat a fly with it, or throw it away, or hang it on the wall. The next chapter will illustrate how these non-standard uses of an object are represented as Precondition indices. These indices do not depend on the functionality of the object, but rather on the states of an actor knowing where an object is, being near it or having possession or control over it. Precondition indices will be handled differently than the strictly functional cases.

2.5 Using Indices For Understanding

The bottom-up indices from GRASP indicate the possible inferences that could be made from the action of GRASPing the New Yorker magazine. That is, they are plans and scripts that the GRASP might be part of. Correspondingly, the TOP-DOWN indices from a goal are the possible plans for achieving that goal. The next chapter, Chapter 3, will list fifteen goals and eleven primitive actions and give the top-down and bottom-up index rules from each of them.

If an understander were to infer all the possible plans listed under the GRASP EXPANDED BOTTOM-UP INDICES, and then INTERSECT this list with the list of TOP-DOWN INDICES from SATISFY-HUNGER, a single intersection results: $RESTAURANT. This process is the basis for the understanding mechanism implemented in ARTHUR. We will see this process outlined in detail in the rest of this chapter.

Assuming for now that this intersection will work as proposed, what good is this result? This alone does not provide a connection between the GRASP action and the SATISFY-HUNGER goal. What it does is to serve to select the path that should be followed in order to connect these. We have said that an explanation index uniquely specifies an inferential path between action and explanation. Hence the $RESTAURANT index completely specifies an inferential path

from GRASP NEW YORKER to SATISFY-HUNGER, since it specifies both the path bottom-up from the GRASP action to the $RESTAURANT index, and the path top-down from the SATISFY-HUNGER explanation to the $RESTAURANT index. Therefore, the understander need not expand the complete chain of inference leading from the action to the goal; by specifying the explanation index, the entire inference chain has been specified, and could be expanded on demand.

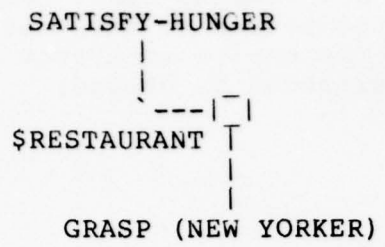
2.5.1 Comparison with PAM

Let us pause a moment to compare ARTHUR's bottom-up inference process with that of PAM. We have said that from the GRASP action, ARTHUR simply infers the set of bottom-up indices from the action: READ, $RESTAURANT and SEE(SHOW). PAM, on the other hand, would have inferred a TAKE(NEW YORKER) plan from the GRASP(NEW YORKER) action, then a DELTA-CONTROL(NEW YORKER) goal from TAKE, then READ(NEW YORKER) from DELTA-CONTROL, then both DELTA-KNOW(LOC SHOW) and DELTA-KNOW(LOC RESTAURANT) (in parallel), then DELTA-PROX(LOC SHOW) and DELTA-PROX(LOC RESTAURANT), respectively, and finally $RESTAURANT and SEE(SHOW).

At this point, PAM will have succeeded in generating precisely the same set of usable plan inferences that ARTHUR generated in a single step: READ, $RESTAURANT and SEE(SHOW). ARTHUR is able to generate these inferences in a single step because ARTHUR contains a precompiled indexed list of the possible bottom-up inferential paths that PAM would generate from any given action. This precompiled list is used as an index to the set of possible bottom-up inferential paths.

Once PAM had generated the above tree of bottom-up inferences, its next bottom-up step from $RESTAURANT will lead to a connection with the SATISFY-HUNGER goal, thereby arriving at an explanation for the GRASP action. As we have seen, ARTHUR has already generated a top-down index from the previously inferred SATISFY-HUNGER goal, containing the indices $RESTAURANT, $GROCER, EAT, FIND(FOOD). The only index in common between the bottom-up index from GRASP and the top-down index from S-HUNGER is $RESTAURANT, and hence this becomes the explanation index for the action-goal pair in this case. An explanation index is the intersection of bottom-up and top-down indices from an action and a goal. Recall that the bottom-up index from the action is an index to the possible bottom-up inferential paths from the action, and the top-down index from the goal is an index to the possible top-down inferential paths from the goal. Therefore, the intersection of the two indices, the explanation index $RESTAURANT, in this case identifies two inferential paths which if followed from the action and goal will result in a connected path from the action to the goal,

via the index $RESTAURANT. We represent this path from action to goal via explanation index by the following connected explanation-graph:



[Figure 6]

Let us briefly outline what we have said so far. There are many possible top-down inference paths from most goals, and many possible bottom-up inference paths from most actions. An understander wishes to find connections between goals and actions in a story, in order to explain the actions in terms of the goals, as described in Chapter 1. Plans and scripts in this scheme are viewed as nothing more than the possible connection points between goals and actions. By INDEXing the possible top-down paths from goals and bottom-up paths from actions all at the level of plans, we can find the intersection between goals and actions. The rest of this chapter will be devoted to showing this process of using indices in understanding.

2.5.2 An Example

Recall the "New Yorker" example:

[20] Willa was hungry. She picked up the New Yorker.

We proposed in a previous section that in order to understand this story, we should first predict all the top-down indices from SATISFY-HUNGER, and then search all the bottom-up indices from DELTA-CONTROL until an intersection is found. In this section we will examine in detail the process of using indexing to understand this story, and present a set of rules which govern the processing described here.

GENERATING A TOP-DOWN INDEX FROM A GOAL.

Consider the first sentence of [20], "Willa was hungry." This statement contains an implicit goal, SATISFY-HUNGER. When the understander reads this sentence, it should infer the implicit SATISFY-HUNGER goal, and predict a set of top-down indices. We express this behavior in the following processing rule:

Rule 6: Implicit statement of a goal

If a story statement contains an implicit goal, then infer that goal from the statement.

In this story, the conceptualization for the sentence implies the goal state SATISFY-HUNGER, so that goal is inferred by Rule 6.

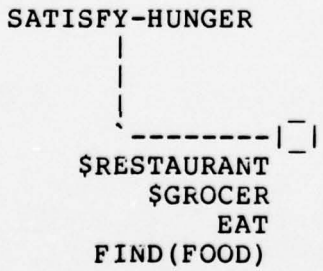
Once ARTHUR has generated this goal, it then generates the top-down index from this goal. This top-down index constitutes the predictive inferences ARTHUR makes from this goal.

Rule 7: Generating predictive inferences

If a goal is inferred in the process of understanding a story,

then generate a set of predictive inferences from that goal, in the form of the top-down index from that goal.

As we saw previously, there are four such predictive inferences from SATISFY-HUNGER. The explanation graph for the example so far contains just the goal SATISFY-HUNGER and the predicted inferences in the form of a top-down index. No actions have yet been mentioned in the story. Hence the explanation graph so far looks as follows:



[Figure 7]

GENERATING BOTTOM-UP INDICES FROM ACTIONS

We have so far generated an explanation of SATISFY-HUNGER for the first sentence, and we have generated a top-down index from that explanation.

Now the second sentence is read: "She picked up the New Yorker magazine". This sentence is analyzed into the following Conceptual Dependency action:

(GRASP ACTOR (WILLA) OBJECT (NEW YORKER) TO (WILLA))

ARTHUR infers the set of bottom-up indices from this action. The bottom-up indices from this action arise from the GRASP itself, and from the fillers of the ACTOR, OBJECT and TO slots. The rules for generating bottom-up indices are given in the next chapter, Chapter 3.

Rule 8: Generating Bottom-Up Indices

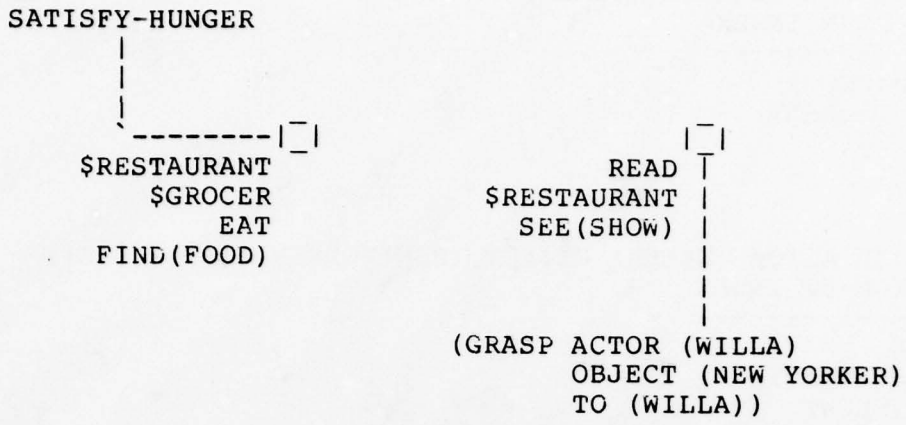
If a story contains a statement of an action,

then infer a bottom-up index from that action via the bottom-up index rules corresponding to the action.

This yields the bottom-up index list given previously for this action:

```
(GRASP ACTOR (WILLA) OBJECT (NEW YORKER) TO (WILLA))
EXPANDED BOTTOM-UP INDICES
--------------------------
$RESTAURANT
SEE(SHOW)
READ
```

Now we have generated a goal and its top-down indices and an action and its bottom-up indices. The explanation-graph for the example so far is as follows:



[Figure 8]

SPECIFICATION SEARCH: INTERSECTING INDICES

The next step in the postdictive understanding process is to attempt to find a connection between the action and the goal: that is, to explain the GRASP action in terms of the previously inferred goal of SATISFY-HUNGER. In order to do this, ARTHUR intersects the two indices, performing what we call a SPECIFICATION SEARCH.

Rule 9: Specification Search

If a character in a story has a goal and has performed an action,

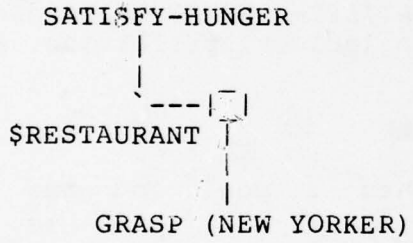
then attempt to explain the action in terms of the goal by performing a specification search, which consists of intersecting the bottom-up index from the action with the top-down index from the goal. The result of this intersection is called the EXPLANATION INDEX.

What is the result of the specification search for the current example? Let us examine the two indices being intersected:

```
(S-HUNGER PLANNER (WILLA))
TOP-DOWN INDEX
--------------
FIND(FOOD)
$RESTAURANT
$GROCER
EAT


(GRASP ACTOR (WILLA) OBJECT (NEW YORKER) TO (WILLA))
BOTTOM-UP INDEX
---------------
READ
$RESTAURANT
SEE(SHOW)
```

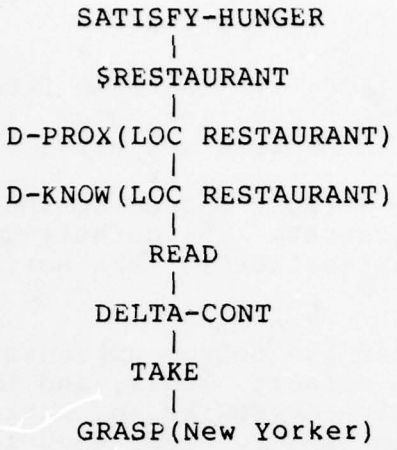
In this case, there is a single member of the intersection of these two sets: $RESTAURANT. This is then the explanation index for this action-goal pair. This explanation index uniquely specifies the appropriate path for the understander to pursue from the GRASP action all the way up to the S-HUNGER goal. The explanation-graph for the complete explanation is as follows:



[Figure 9]

Recall that the explanation index uniquely specifies the inferential path from the action up to the index and the path from the goal down to the index. Since both paths meet at the explanation index, a complete path from action to goal has been uniquely specified.

If the above explanation-graph were expanded to instantiate all the inferences from the GRASP action up through the $RESTAURANT index and on up to the SATISFY-HUNGER goal, the entire inference chain would look as follows:



[Figure 10]

2.6 More examples, more rules

We have introduced three understanding rules which have used indices to arrive at an explanation for example [20]. In this section, we will expand the scope of the above rules to account for some more complex stories.

2.6.1 Generating a Default Explanation

Consider what would happen if the "New Yorker" story were reversed, i.e., if the GRASP NEW YORKER action was the first sentence, as follows:

[22] Willa picked up the New Yorker. She was hungry.

In this reversal of the Willa story, the first sentence does not explicitly contain a goal. What is the understander to infer from this sentence? We have seen that ARTHUR infers the complete set of bottom-up inferences from actions in a story. However, if a person reads this first sentence, he is able to answer why Willa might have picked up the New Yorker magazine; namely, she probably wanted to read it. Hence, we propose that in the absence of any previous applicable goal information in a story, an understander generates a default goal explanation for actions in the story.

Rule 10: Default Explanations

If a bottom-up index is generated from an action, and there is no existing goal in the story representation to explain the action,

then choose the default bottom-up index from the action, and generate the default goal from that index as the explanation for the action.

This rule assumes that the bottom-up index from an action contains a marked default index, and that that index has stored a default goal. ARTHUR has a marked default index for every bottom-up index, and each index has a default goal. Since some actions' indices depend on the functionality of the filler of the OBJECT slot in the action, the functional definitions of objects must include a definition of the default function indices. Some examples of default function indices in ARTHUR's memory are:

OBJECT	DEFAULT FUNCTION INDEX
GUN:	THREATEN
NOVEL:	READ
COOKBOOK:	$COOK
MICH.GUIDE:	$RESTAURANT
MAP:	FIND(LOC)
MUSEUM:	$MUSEUM
SANDWICH:	EAT
CAR:	$DRIVE

2.6.2 Constraining Top-Down Indices

Recall that we have said that ARTHUR generates the top-down index from any goal it inferred while understanding a story. For example, from the SATISFY-HUNGER goal in the original "New Yorker" example, ARTHUR generated the top-down index of four plans and scripts from that goal. It is not always the case that all the known top-down indices from an inferred goal will be generated. In that example, the SATISFY-HUNGER goal is generated directly from the statement of Willa's being hungry. However, consider what would happen if the first sentence of a story was the following:

[23] John went to a restaurant.

n this case, the action of PTRANSing to a RESTAURANT causes the functional attributes of RESTAURANT to be looked up, yielding a bottom-up index of $RESTAURANT. Since we have

said that this is tne first sentence of a story, a default goal is inferred according to rule 10. In this case the goal will be SATISFY-HUNGER. In this case, however, we will NOT generate the complete list of four top-down indices as predictions from this SATISFY-HUNGER inference. If we did, that would be equivalent to predicting that John might use any one of these four possible plans to satisfy his hunger, whereas in fact we know from the first sentence that he has already selected a plan: going to a restaurant, i.e., doing the $RESTAURANT script. Hence if we wish to make any predictions from the SATISFY-HUNGER goal, we want to predict that the $RESTAURANT script will be used. This accounts for the fact that an understander would notice the change in plan if John decided to go to the grocery store instead, as in the following example:

[24] John went to a restaurant. He decided to go to the grocery store.

Once the $RESTAURANT script has been inferred as being in service of SATISFY-HUNGER in this example, then the $GROCER script is no longer a valid prediction from the SATISFY-HUNGER goal. Hence in example [24] we recognize that John must be employing a new plan for satisfying his hunger, and that the RESTAURANT script is no longer being used. We propose the following rule:

Rule 11: Constraining Top-Down Indices (1st Cut)

If a theme-level goal is the explanation for an action via a specific explanation index,

then the top-down predictive index generated subsequently from that goal should consist of just the explanation-index itself. The resulting top-down index is called a <u>constrained</u> top-down index.

If a story contains an implicit goal statement,

then the top-down index from that goal is unconstrained; i.e., the complete set of top-down indices should be generated from that goal.

Hence we see that in examples [23] and [24], a constrained top-down index should be predicted: this constrained top-down index will consist of the single script $RESTAURANT, since that is the single explanation index for the first sentence of the each example. However, in the case of the original New Yorker story, example [20], an unconstrained top-down index will be generated, consisting of the four top-down indices given previously for SATISFY-HUNGER, since the goal is implicitly stated in the statement that "Willa was hungry."

2.6.3 Compatible indices and the PERSUADE package plans

Consider [25]:

[25] John wanted to borrow bill's bicycle. He asked Bill to lend it to him. Bill refused. Then John offered to do Bill's laundry if Bill would lend him the bike.

Schank and Abelson [1977] pointed out that the plans in the PERSUADE package can only be used as an ordered set; i.e., they can only be used in the following order, given by Schank and Abelson: ASK, INVOKE THEME, INFORM REASON, BARGAIN OBJECT, BARGAIN FAVOR, THREATEN, OVERPOWER, STEAL. Hence story [25] above makes sense: John first tried the ASK planbox, and then he tried the BARGAIN-FAVOR planbox.

If, after the ASK plan is used, we constrain the top-down index from the ACHIEVE-POSSESSION(BIKE) goal, we will then fail to match the subsequent BARGAIN-FAVOR plan. What we should have done instead is to constrain the top-down index to be only those plans which come after ASK in the PERSUADE package.

Now consider [26]:

[26] John wanted to borrow Bill's bicycle. He offered to do Bill's laundry if Bill would lend it to him. Bill refused. Then John asked Bill to lend the bike to him.

This story seems odd, since John first tried the BARGAIN-FAVOR plan and then tried the ASK plan. Understanders have knowledge that says that if a given PERSUADE plan fails, one of the subsequent plans in the list might still work, but none of the previous ones will.

If, after the BARGAIN-FAVOR plan, we constrain the top-down index from the ACHIEVE-POSSESSION(BIKE) goal to be only those plans which follow BARGAIN-FAVOR in the PERSUADE package, then we will have ruled out the ASK and a few other plans. Hence the use of the ASK plan will initially fail to be explained by the constrained top-down index. This corresponds to the fact that people consider this use of the ASK plan to be anomalous.

In the above examples, the BARGAIN-OBJECT plan is a valid plan in service of the ACHIEVE-POSSESSION goal, even when that goal has a constrained top-down index of the ASK plan. The reverse is not true: the ASK plan is not a valid inference from the ACHIEVE-POSSESSION goal if that goal has a constrained top-down index of BARGAIN-OBJECT. We call plans that are valid inferences from a goal even though they

are not in the constrained top-down index from that goal "compatible indices".

2.6.4 Conjunctive plans

There are goals which require more than one plan to be implemented in order to achieve the success of that goal. For example, consider [27]:

> [27] John wanted to get his Ph.D. He took the required courses. Then he wrote a thesis.

After John's action of taking courses in service of his goal of getting the Ph.D., we would constrain the top-down index from that goal to include that action of taking courses AND the action of writing a thesis. These actions would be listed in the understanders memory as both being necessary for the goal, rather than as being alternative plans for achieving the goal. Such a set of plans is called a set of conjunctive indices.

We now modify our rule about constraining top-down indices to read as follows:

> Rule 11: Constraining Top-Down Indices
>
> If a theme-level goal is the explanation for an action via a specific explanation index,
>
> then the top-down predictive index generated subsequently from that goal should consist of just the explanation-index itself, plus indices which are compatible or conjunctive indices to the explanation-index. The resulting top-down index is called a constrained top-down index.
>
> If a story contains an implicit goal statement,
>
> then the top-down index from that goal is unconstrained; i.e., the complete set of top-down indices should be generated from that goal.

2.7 Precondition indices

Consider the following:

> [28] Mary wanted something to throw in a long and curving arc. She picked up her tennis racquet.

Understanding this example requires the reader to infer that Mary was going to throw the racquet in a long arc. The problem is that this is not a known function of a tennis racquet. If we included "being thrown in a long and curving arc" as a bottom-up index from getting a tennis racquet, then we would essentially have an infinite number of possible indices from every possible action. This would mean that searching through the indices would be impossible due to combinatorial explosion, and all of the benefits that we have claimed for indexed inference paths would be lost.

We clearly cannot infer indices like "throwing in a long arc" as bottom-up indices from the action of getting a tennis racquet. However, an understander must be able to recognize a use like that in [28] when it occurs in a story. This is a potential dilemma: the understander must be able to recognize virtually any use of an object, but the understander must not explicitly infer non-standard uses ahead of time. This problem can be expressed as follows:

> An understander must recognize non-standard uses of an object,
>
> but the understander must not explicitly infer such uses of an object.

The key to the solution of this paradox is the word "explicit". What the understander must do is <u>implicitly</u> infer the non-standard uses of an object. What we mean by this is that in addition to the known functions of an object, the understander infers a "catch-all" index which specifies that the object may be used in some non-standard way. Then any plan in which the object appears can be inferred to be a non-standard plan for that object.

This catch-all index can be adapted from Schank and Abelson's [1977] notion of <u>preconditions</u>. For example, they say that in order to perform a plan which requires a gun, an actor must first get control of a gun. This is true <u>even if the plan involves a non-standard use of the gun</u>. Hence, in [28], we infer that Mary's plan involves some object, and we then infer that the tennis racquet can be that object.

Notice that we must make two inferences here: one, that Mary plans to throw an object to satisfy her goal, and two, that the tennis racquet can be that object. The second inference can be constrained by specifying some features of the object that is to be used in the non-standard plan. For example, consider [29]:

[29] John wanted something to brush his teeth with. He picked up his bicycle.

Most readers find themselves attempting to infer that John is about to brush his teeth with his bicycle somehow, but they also know that that is an absurd inference, in that a bicycle can't be used in that way. Similarly, consider the following:

[30] Mary wanted to swat a fly. She picked up a glass of water.

Again, the story seems to imply that Mary is somehow going to swat the fly with a glass of water, but we know that a glass of water is an inappropriate instrument with which to swat a fly.

Specifying the "appropriate" features of a toothbrush or a flyswatter is a very difficult task. For example, consider the following list of possible flyswatting instruments:

a piece of heavy fabric
a shoebox
a chalk eraser
your hand
a sandwich

These aren't all normal flyswatters, but we can envision using any of them as such. However, the following cannot be so envisioned:

a sheet of paper
a tinderbox
a pencil eraser
the back of your head
a plate of spaghetti

An understander needs a very good taxonomy of features of physical objects in order to adequately specify the rules by which we decide the appropriateness of using a certain object for a certain task. Without such a taxonomy of object features, an understander might infer a use of an object that a person would not infer, or will fail to see a

possible use that a person would see.

Putting aside the problem of object features, we can see that if an understander has some basis on which to infer that an object might play some role in a plan, then the action of getting that object can be inferred to be a precondition for the plan. In addition, finding out the location of the object, or going to that location can also be considered preconditions for a plan which makes some (non-standard) use of the object. These three preconditions are called DELTA-CONTROL, DELTA-KNOW and DELTA-PROX by Schank and Abelson.

Rule 14: Precondition inferences

If an object fills some role in a plan,

then the actions of finding out the location of the object (MTRANS, ATTEND, MBUILD), going to that location (PTRANS), or getting control of the object (ATRANS, GRASP) can be interpreted as satisfying, respectively, DKNOW, DPROX or DCONT preconditions for that plan.

Preconditions are used in ARTHUR for understanding non-standard uses of objects. Whenever one of the above six CD ACTs occurs in a story, in addition to inferring any standard uses of the objects included in the action, ARTHUR infers the appropriate precondition index corresponding to the act, according to Rule 14. Then if specification search fails to find an explanation index using the standard indices from the action, ARTHUR checks the precondition index against the top-down indices from the goal. This search is more flexible than specification search: if any top-down index can account for all the elements of the precondition index, then ARTHUR infers that that the action may have been performed as a precondition in service of the goal.

Rule 15: Preconditions in understanding

If specification search fails to yield an explanation index between an action and a goal,

and a precondition index from the action can be accounted for by a top-down index from the goal,

then infer that the action may satisfy a precondition for that top-down index in service of the goal.

2.8 Loose-end explanations

Consider the following examples:

[31] John whispered something to Mary.

[31] Sue picked up a blade of grass.

[31] Jack went to Boston.

Each of these statements consists of a CD action which specifies too little to base a solid inference on. An MTRANS without an MOBJECT, an ATRANS of an object which has no known function, a PTRANS to a location with no particular associated script. Consider the last of these, "Jack went to Boston." If we ask a reader "Why did Jack go to Boston", the reader will have no specific default answer, as he would if Jack went to a restaurant, or to a baseball game. The best answer a reader can give is that Jack probably went because he had something he wanted to do there. This is equivalent to saying that Jack's going to Boston was a precondition for some as-yet-unspecified plan.

In other words, the default bottom-up inference from this PTRANS action is the DPROX precondition inference, which is so general that it can match virtually any plan in which Boston plays some part. Whenever such an action is read in a story, the understander does not infer a default goal for that action, but only infers the bottom-up precondition index, as yet unattached to any goal. We call such an unattached bottom-up index a loose-end explanation, denoting the fact that the explanation is still incomplete and must be attached to some subsequent goal in order to be fully explained.

Rule 16: Leaving a loose end

If a statement specifies an action without specifying all of the important rolefillers of that action,

Then do not infer a default explanation for the action, but instead leave a "loose-end" explanation, consisting of the bottom-up index from the action.

The "important rolefillers" of an action are specified by the following table:

CD ACT	IMPORTANT ROLEFILLERS
MTRANS	MOBJECT
PTRANS	TO
ATRANS	OBJECT
PROPEL	OBJECT, TO
INGEST	OBJECT
EXPEL	OBJECT
ATTEND	TO
MBUILD	MOBJECT
SPEAK	MOBJECT
MOVE	OBJECT
GRASP	OBJECT

This means, for example, that if a PTRANS is specified without a destination, then ARTHUR will leave a loose-end explanation for it. ARTHUR can perform specification search between a loose-end explanation and the top-down index from a subsequent goal. If the loose end is found to be a precondition index for some subsequent goal by this method, we say the loose end has been resolved into a full explanation. More will be said about loose end explanations in subsequent chapters.

2.9 ARTHUR output

Following is an annotated run of ARTHUR on a simple example which involves finding a known connection between a goal and an action by indexing.

```
[PH: Initiation. 16-Oct-79  7:31AM]

 Yale TOPS-20 Command Processor 3A(415)-2

@RUN ARTHUR

*(ARTHUR)

Input story: *EX2

The story is:

 JOHN WANTED MONEY.
 HE SOLD A WATCH.
```

ARTHUR OUTPUT	ANNOTATION
The story is: JOHN WANTED MONEY. HE SOLD A WATCH. Conceptual Dependency: ((A-POSSESS (ACTOR JOHN) (OBJECT MONEY)) (MUTUAL-ATRANS (ACTOR JOHN) (OBJECT WATCH) (FOR MONEY))) *** ARTHUR: UNDERSTANDING PHASE -------------- NEXT SENTENCE: -------------- JOHN WANTED MONEY. *** INPUT IS GOAL: (A-POSSESS (ACTOR JOHN) (OBJECT MONEY)) NEW EXPLANATION (A-POSSESS (ACTOR JOHN) (OBJECT MONEY)) ADDED TO EXPLANATION-GRAPH UPDATING EXPLANATION-GRAPH: EXPLANATION TRIPLE: GOAL: ((A-POSSESS (ACTOR JOHN) (OBJECT MONEY))) EVENT: (A-POSSESS (ACTOR JOHN) (OBJECT MONEY)) INDEX: NIL STATUS: ACTIVE	ARTHUR prints the Conceptual Dependency representation of the story. First sentence recognized as a statement of a goal: A-POSSESS. A-POSSESS is a theme-level goal, so no further explanation of it is necessary. The syntax of explanation graphs is that they state a goal followed by all events explained by that goal, and the explanation index for each event under each goal. In this graph, the story stated a goal directly, so the goal is given as the explanation for itself.

```
GENERATING TOP-DOWN INDEX:
((ASK
  (ACTOR JOHN) (OBJECT MONEY))
 (INVOKE-THEME
  (ACTOR JOHN) (OBJECT MONEY))
 (INFORM-REASON
  (ACTOR JOHN) (OBJECT MONEY))
 (BARGAIN-OBJECT
  (ACTOR JOHN) (OBJECT MONEY))
 (BARGAIN-FAVOR
  (ACTOR JOHN) (OBJECT MONEY))
 (THREATEN
  (ACTOR JOHN) (OBJECT MONEY))
 (OVERPOWER
  (ACTOR JOHN) (OBJECT MONEY))
 (STEAL
  (ACTOR JOHN) (OBJECT MONEY))
 (PLAY-GAME
  (ACTOR JOHN)
  (COMPETITION WAGER)))
FOR GOAL:
(A-POSSESS (ACTOR JOHN)
           (OBJECT MONEY))

--------------
NEXT SENTENCE:
--------------
 JOHN SOLD A WATCH.

*** INPUT EVENT:
    (MUTUAL-ATRANS
     (ACTOR JOHN)
     (OBJECT WATCH)
     (FOR MONEY))

GENERATING BOTTOM-UP INDEX:
((DELTA-CONT
  (ACTOR JOHN) (OBJECT MONEY))
 (BARGAIN-OBJECT
  (ACTOR JOHN) (OBJECT MONEY)))
FOR EVENT:
(MUTUAL-ATRANS (ACTOR JOHN)
               (OBJECT WATCH)
               (FOR MONEY))
```

Once the goal has been inferred, a top-down index is generated, as a prediction of what plans might be performed in service of the goal. In this case, the top-down index consists of the plans in the PERSUADE package, along with the plan of wagering.

This story input is an event, interpreted as John performing a MUTUAL ATRANS: he ATRANSes the watch to someone and they ATRANS money to him in return.

The bottom-up index from the event consists of two plans: John wants to have money, or else John wants to persuade someone to give him money by bargaining.

```
PERFORMING SPECIFICATION SEARCH

FOUND EXPLANATION INDEX:
(BARGAIN-OBJECT
 (ACTOR JOHN) (OBJECT MONEY)))

UPDATING EXPLANATION-GRAPH:

EXPLANATION TRIPLE:

 GOAL: ((A-POSSESS
         (ACTOR JOHN)
         (OBJECT MONEY)))

   EVENT: (A-POSSESS
           (ACTOR JOHN)
           (OBJECT MONEY))
   INDEX: NIL
   STATUS: ACTIVE

   EVENT: (MUTUAL-ATRANS
           (ACTOR JOHN)
           (OBJECT WATCH)
           (FOR MONEY))
   INDEX: ((BARGAIN-OBJECT
            (ACTOR JOHN)
            (OBJECT MONEY)))
   STATUS. ACTIVE

*** END OF STORY
*** READY FOR QUESTIONS
```

The top-down index from A-POSSESS is intersected with the bottom-up index from MUTUAL-ATRANS. The intersection process yields a single explanation index, the plan BARGAIN-OBJECT.

The A-POSSESS goal now serves as explanation for the MUTUAL-ATRANS action, via the BARGAIN-OBJECT index. In other words, the MUTUAL-ATRANS is interpreted as having been performed in service of the goal of A-POSSESS money.

```
Input question:*EX2Q1

QUESTION:
 WHY DID JOHN SELL A WATCH?

ANSWER:
 BECAUSE HE WANTED TO GET MONEY.

Input question:*EX2Q2

QUESTION:
 WHY DID JOHN WANT MONEY?

ANSWER:
 THE STORY SAID THAT HE WANTED SOME.

[PH: Termination. 16-Oct-79  7:32AM. PS:<GRANGER.TH>WATCH.LOG.1]
```

2.10 Summary: Indexing Explanations

We have described a process whereby a connection between an action and a goal can be found without generating all possible bottom-up inferences from the action in parallel, and without predicting a large number of top-down inferences from the goal. This was successful even though the process realistically took into account the many alternate possible paths that could have arisen from the action and the goal. This process was accomplished by the process of indexing explanations. The following six processing rules were introduced:

Rule 6: Implicit statement of a goal

If a story statement contains an implicit goal, then infer that goal from the statement.

Rule 7: Generating predictive inferences

If a goal is inferred in the process of understanding a story,

then generate a set of predictive inferences from that goal, in the form of the top-down index from that goal.

Rule 8: Generating Bottom-Up Indices

If a story contains a statement of an action,

then infer a bottom-up index from that action via the bottom-up index rules corresponding to the action.

Rule 9: Specification Search

If a character in a story has a goal and has performed an action,

then attempt to explain the action in terms of the goal by performing a specification search, which consists of intersecting the bottom-up index from the action with the top-down index from the goal. The result of this intersection is called the EXPLANATION INDEX.

Rule 10: Default Explanations

If a bottom-up index is generated from an action, and there is no existing goal in the story representation to explain the action,

then choose the default bottom-up index from the action, and generate the default goal from that index as the explanation for the action.

Rule 11: Constraining Top-Down Indices

If a theme-level goal is the explanation for an action via a specific explanation index,
then the top-down predictive index generated subsequently from that goal should consist of just the explanation-index itself, plus indices which are compatible or conjunctive indices to the explanation-index. The resulting top-down index is called a constrained top-down index.

If a story contains an implicit goal statement,
then the top-down index from that goal is unconstrained; i.e., the complete set of top-down indices should be generated from that goal.

The explanation process described by these rules took us through several examples, finding a connection between actions and goals without requiring backup nor any parallel bottom-up inferences. The only search mechanism employed

was that of specification search through two lists of indices.

We have so far investigated only simple goal-based stories in which the goal is stated before the action. We shall see that there are a great many more processing rules that will be necessary to account for more complex stories. The rest of this thesis presents progressively more complex stories and introduces progressively more complex processing rules to account for ARTHUR's processing of these stories.

2.11 End of one chapter, beginning of the next.

We have designed a process whereby connections between goals and actions can be found by indexing the possible top-down and bottom-up inferential paths that can be followed from goals and actions. Only these indices need be generated from actions and goals, rather than the complete set of parallel inference chains that were required in earlier explanation-driven understanding processes.

We have seen some examples of top-down and bottom-up indices from goals and actions, but we have not yet seen how these indices are generated or where they come from. The next chapter, Chapter 3, presents the rules for generating bottom-up and top-down indices from actions and goals.

CHAPTER 3

THE INDEXED INFERENCES OF ACTIONS AND GOALS

3.1 Introduction

The previous chapter presented a method of finding connections between actions and goals. This was done by generating bottom-up indices from actions and top-down indices from goals, and intersecting the two to yield a single index. That single index specified the inferential path which would lead from the action to the goal. In this chapter, we present the indices that are generated from actions and goals.

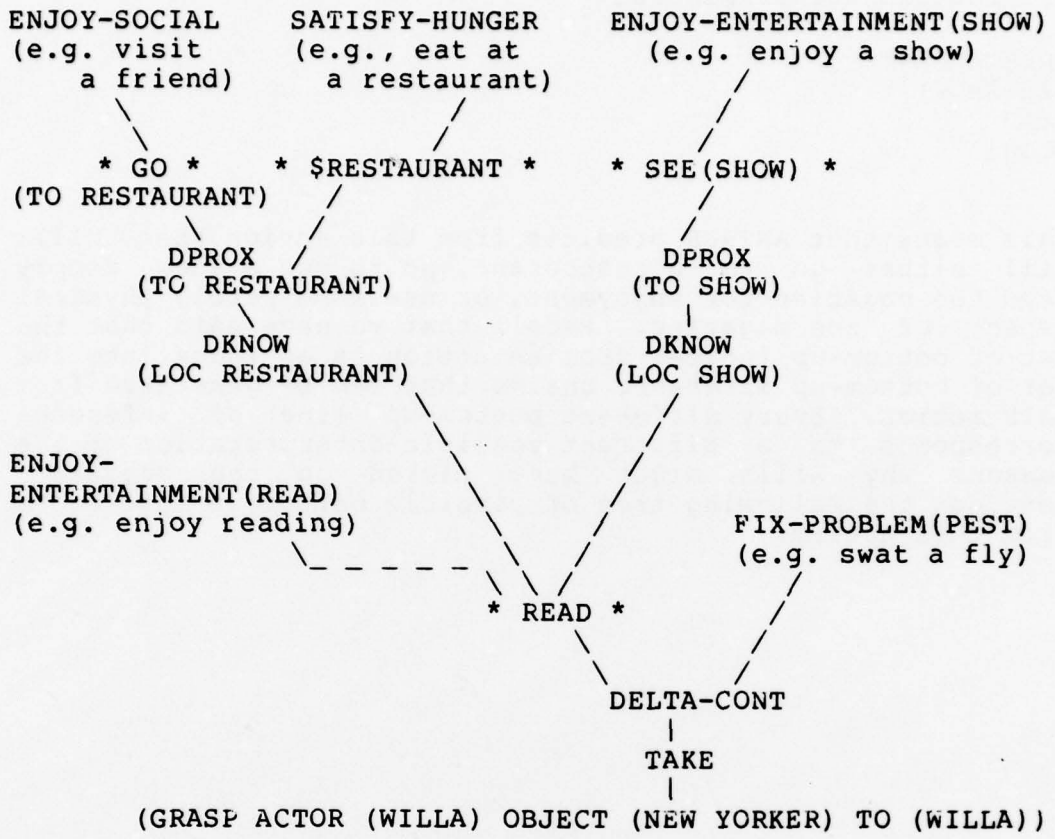
We have already seen that the indices from an action depend not only on the action itself, but on the fillers of the ACTOR, OBJECT and TO slots in the instantiated action. For example, if Willa picks up a gun, we do not generate our bottom-up inferences solely on the basis of her GRASP action, but also according to what we know about guns and their uses, and Willa and her intentions. Hence the bottom-up indices from an action are generated by rules which instruct ARTHUR where to look in its memory, according to the slotfillers of an action.

Consider the action of Willa picking up the New Yorker magazine. We saw in the previous chapter that the set of bottom-up indices from that action was as follows:

(GRASP ACTOR (WILLA) OBJECT (NEW YORKER) TO (WILLA)):
EXPANDED BOTTOM-UP INDICES

$RESTAURANT
SEE(SHOW)
READ
DCONT

This means that ARTHUR predicts from this action that Willa will either go to a restaurant, go to see a show, simply read the magazine for enjoyment, or use some purely physical aspect of the magazine. Recall that we have said that the set of bottom-up indices from an action is an index into the set of bottom-up inference chains that can be generated from this action. Every different bottom-up line of inference corresponds to a different possible interpretation of the reasons why Willa might have picked up the magazine. Consider the following tree of possible bottom-up inferences from this action:



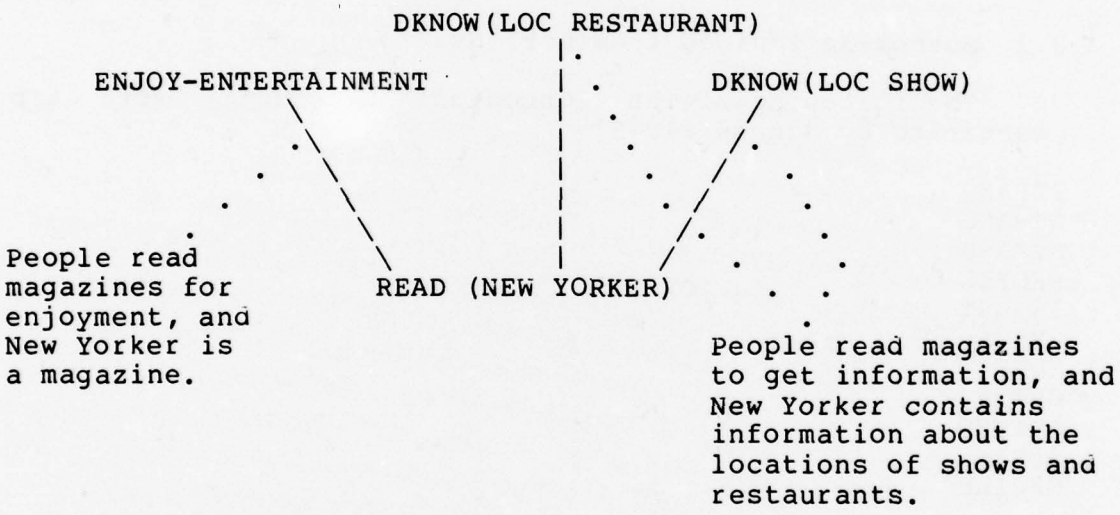
[Figure 11]

Consider an understander that generates this tree, step by step, from the action of GRASPing the New Yorker. Consider in particular the bottom-up inference from READ. At this step, there are three possible paths to follow, leading up to DKNOW(LOC RESTAURANT), DKNOW(LOC SHOW) and ENJOY-ENTERTAINMENT. This means that once the understander has inferred that Willa might want to READ the magazine, the next step is to infer exactly WHY Willa might want to read it.

The understander has a rule that says that people read things in order to get information about them, or simply for enjoyment, with no higher aim in mind. The New Yorker magazine is a source of two different kinds of information (at least). For the purposes of this illustration, we just focus on two kinds of information in the New Yorker: information about the locations of restaurants and

information about locations of shows. The understander knows that a New Yorker contains these two types of information. Hence at the point that the understander infers what kind of information Willa might want to know, it looks in its memory entry for the New Yorker magazine, and pulls out the information that the magazine contains info about restaurants and shows. This knowledge about the New Yorker is the knowledge that is responsible for the understander's generating the two branches of the bottom-up inference tree: DKNOW(LOC RESTAURANT) and DKNOW(LOC SHOW). The third branch, ENJOY-ENTERTAINMENT, is due to the understander's knowledge that any book or magazine may be read for enjoyment.

The following diagram depicts the three inferences from READ(NEW YORKER) along with the information the understander used to generate those inferences:



This diagram illustrates that the understander is applying its rules about why people read to specific attributes of the New Yorker. This is the case for every inferential step: at each inferential step, the understander chooses a particular attribute of one of the elements of the action: GRASP, WILLA and NEW YORKER.

Instead of only making use of one piece of information in the action at a time to generate step-by-step bottom-up inferences, ARTHUR looks up all the information contained under each of the parts of an action and generates the bottom-up indices that are pre-stored there. We hypothesize that since an understander can generate the entire tree shown in Figure 11 on demand, it could instead have the

paths of that tree stored under the appropriate elements of the action itself. The "appropriate" place to store a given piece of information simply means the slotfiller that would have given rise to the information in a step-by-step understander.

This chapter presents the rules for generating bottom-up indices from the eleven primitive actions of Conceptual Dependency theory. We will see that these rules depend on the slotfillers of an instantiated action, just as they would for a step-by-step inferencer. The difference is that in ARTHUR, the tree that would result from a step-by-step analysis is pre-stored under the actions and objects known to ARTHUR, and this pre-stored information is generated all at once from an action. The information generated in this manner comprises the bottom-up index from an action.

3.2 Bottom-Up Indices from Actions

The following eleven Conceptual Dependency acts are identified by Schank [1975]:

PTRANS
ATRANS
MTRANS
PROPEL
INGEST
EXPEL
MOVE
GRASP
ATTEND
SPEAK
MBUILD

The following table gives the BOTTOM-UP INDICES for each of the 11 Conceptual Dependency actions in the above list. Notice that the table refers to indices which are rules specifying particular memory attributes of fillers of the ACTOR, OBJECT and TO slots in the instantiated action.

ACT	BOTTOM-UP INDEX
PTRANS	LOCATION-INDICES(TO SLOT) + PLANNER(ACTOR SLOT) + DPROX(ACTOR,TO)
ATRANS	FUNCTIONS(OBJECT SLOT) + PLANNER(ACTOR SLOT) + DCONT(ACTOR,OBJECT)
INGEST	FUNCTIONS(OBJECT SLOT) + PLANNER(ACTOR SLOT)
EXPEL	BREATHE
MOVE	LOCATION-INDICES(TO SLOT) + PLANNER(ACTOR SLOT) + FUNCTION(TO SLOT) + GESTURE(If OBJECT = HAND)
GRASP	FUNCTIONS(OBJECT SLOT) + PLANNER(ACTOR SLOT) + DCONT(ACTOR,OBJECT)
ATTEND	INFORMATION(TO SLOT) + PLANNER(ACTOR SLOT) + DKNOW(ACTOR,TO)
MBUILD	INFORMATION(OBJECT SLOT) + PLANNER(ACTOR SLOT) + DKNOW(ACTOR,MOBJECT)
MTRANS	Any of the plans in the PERSUADE package + TELL (If ACTOR ~= TO and INST = SPEAK) + READ (If TO = EYES of ACTOR) + INFORMATION(MOBJECT SLOT) + PLANNER(ACTOR SLOT) + DKNOW(TO,MOBJECT)
SPEAK	Any of the plans in the PERSUADE package + TELL + PLANNER(ACTOR SLOT)
PROPEL	CHANGE-PHYS-STATE (NEG) + PLANNER(ACTOR SLOT)

There is more than one bottom-up index rule for each action, corresponding to the different inferences we generate depending on the various slotfillers in a conceptualization. For most of the actions, the different rules correspond directly to different slots in the instantiated action. These rules correspond to ARTHUR's implementation of them: that of looking up the attributes of the slotfillers in the instantiated action. In the following discussion, we will present alternate expressions of some of these rules, to give a feel for the knowledge embodied in these rules. We will discuss first the OBJECT and TO slot rules, and then we will discuss the ACTOR slot rules.

3.2.1 The OBJECT and TO slots of actions

Consider the rules given above for generating bottom-up inferences from a PTRANS action. The first rule says "LOCATION-INDICES(TO SLOT)". This rule can be expressed as follows:

Rule 23: Location indices

If an actor PTRANSes to a location which has a stereotypical activity associated with it,

Then infer that the action may be part of a plan to perform that stereotypical activity.

This means that when an instantiated PTRANS action occurs in the representation of a story, ARTHUR looks at the filler of the TO slot in the action, and examines the LOCATION-INDICES attribute of that filler in its memory. ARTHUR has LOCATION-INDICES attributes stored in its memory entries for certain locations. For example, consider the following sentence:

[34] "John went to the racetrack."

This statement is represented in Conceptual Dependency (see Schank [1975]) as a PTRANS with the ACTOR and OBJECT both being John, and the TO slot being the racetrack, as follows:

(PTRANS ACTOR (JOHN) OBJECT (JOHN) TO (RACETRACK))

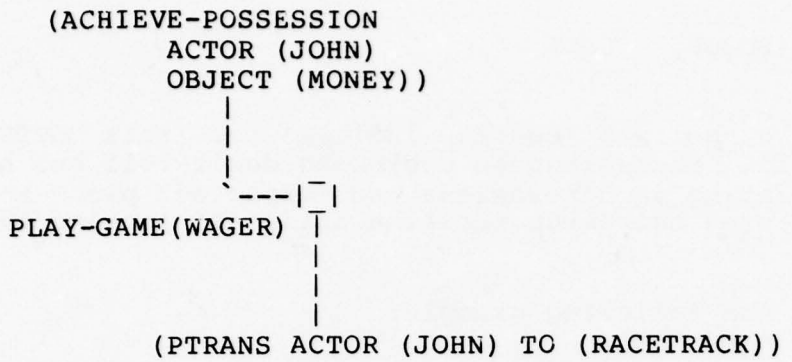
Once this PTRANS action has been represented, the understander performs the bottom-up index rules from this instantiated action. In this case, the TO slot is filled with RACETRACK. ARTHUR's conceptual definition for the concept RACETRACK has the attribute LOCATION-INDICES defined as follows:

RACETRACK
 LOCATION-INDICES (SEE(HORSERACE), PLAY-GAME(WAGER))

This slotfiller has two LOCATION-INDICES. Hence they both become bottom-up indices for this PTRANS action. This means that these indices are the inferences ARTHUR makes from the fact that the racetrack was John's destination. To see how these bottom-up indices affect the explanation of a story, consider the following:

[35] John went to the racetrack. He wanted to win some money.

The bottom-up indices from the first sentence are the two location-indices given above, SEE(HORSERACE) and PLAY-GAME(WAGER). The second sentence states a goal, namely possessing money. This goal then serves as the explanation for the action of going to the racetrack, via the PLAY-GAME(WAGER) index, as in the following explanation graph:



[Figure 12]

Now consider the actions MTRANS, MBUILD and ATTEND. They all have in common the bottom-up inference rule INFORMATION(OBJECT SLOT). This rule can be expressed as follows:

Rule 24: Information indices

If a character is the RECIPIENT of an MTRANS, MBUILD or ATTEND action concerning an object which has a plan associated with it in memory,

Then infer that the action may be part of a plan to perform the associated plan.

This says to look at the INFORMATION attribute of the filler of the OBJECT slot in the instantiated MTRANS, MBUILD or ATTEND action. For example, consider the following sentence:

[36] John was looking in his cookbook.

This sentence is represented as an ATTEND of John's eyes towards the cookbook, as follows:

(ATTEND ACTOR (JOHN) OBJECT (EYES) TO (COOKBOOK))

The INFORMATION attribute under the definition for COOKBOOK in ARTHUR's memory looks as follows:

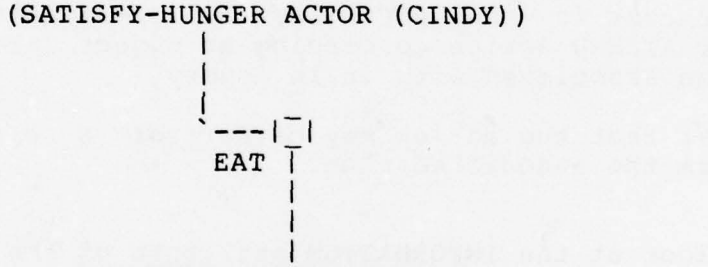
COOKBOOK
 INFORMATION ($COOK, EAT)

Hence these two scripts become indices for this ATTEND action. Notice that although cookbooks don't tell one how to eat, that eating is nonetheless an expected plan that should be inferred bottom-up from the action of looking in a cookbook.

Consider the following example:

[37] Cindy was hungry. She looked in her cookbook.

The first sentence contains a SATISFY-HUNGER goal, and the second sentence mentions an ATTEND EYES to COOKBOOK action. ARTHUR finds a connection between the goal and the action via the EAT plan contained in the bottom-up index generated from the INFORMATION attribute of COOKBOOK. The explanation graph for this example is as follows:



[Figure 13]

Here is some abbreviated output of ARTHUR understanding this example, presented to give a feel for ARTHUR's use of indices in understanding:

```
---------------
*** INPUT GOAL:
---------------
(S-HUNGER (PLANNER CINDY))

-------------------------------------
GENERATING TOP-DOWN PREDICTIVE INDEX:
-------------------------------------
```

```
((EAT (PLANNER CINDY) (FOOD NIL))
 ($RESTAURANT (PLANNER CINDY))
 ($GROCER (PLANNER CINDY))
 (FIND (PLANNER CINDY) (FOOD NIL)))

----------------
*** INPUT EVENT:
----------------
(ATTEND (ACTOR CINDY) (OBJECT EYES (PART CINDY)) (TO COOKBOOK))

---------------------------
GENERATING BOTTOM-UP INDEX:
---------------------------
(($COOK (PLANNER CINDY) (FOOD NIL))
 (EAT (PLANNER CINDY) (FOOD NIL)))

PERFORMING SPECIFICATION SEARCH



------------------------
FOUND EXPLANATION-INDEX:
------------------------
((EAT (PLANNER CINDY) (FOOD NIL)))

UPDATING EXPLANATION GRAPH
```

To summarize, every Conceptual Dependency action has a set of rules for generating bottom-up indices. Each rule refers to the attributes of at least one filler of one of the slots in the instantiated action. Every object known to ARTHUR has certain attributes associated with it in ARTHUR's memory. Among those attributes are FUNCTION and LOCATION-INDICES. Stored under those attributes of objects in ARTHUR's memory are lists of the scripts and plans that are to be generated as part of the bottom-up index for the action that causes them to be accessed.

3.2.2 The ACTOR slot of actions

All of the actions in the above bottom-up index table have the same rule referencing the ACTOR slot: the rule says to look up the indices under the PLANNER attribute of the filler of the ACTOR slot in the action. This rule can be stated as follows:

Rule 25: Inferring a known plan

If a character has a known plan,
and that character performs an action,

Then infer that the action may be part of the plan.

Consider the PTRANS action, for example. Consider the following sentence:

[38] "John went to Connecticut."

This is a PTRANS action, where the ACTOR of the PTRANS is John:

(PTRANS ACTOR (JOHN) OBJECT (JOHN) TO (CONNECTICUT))

What does ARTHUR have in its memory entry for John, under the attribute PLANNER? The answer is nothing: no information. ARTHUR has never heard of John before, just as we have never heard of him before either. Hence neither we or ARTHUR have any special PLANNER information stored for him, as opposed to an object like a racetrack, which an understander does know the FUNCTIONs of. Now consider the following:

[39] "President Carter went to Connecticut."

Again, this is a PTRANS action, this time with the ACTOR being President Carter:

(PTRANS ACTOR (CARTER) OBJECT (CARTER) TO (CONNECTICUT))

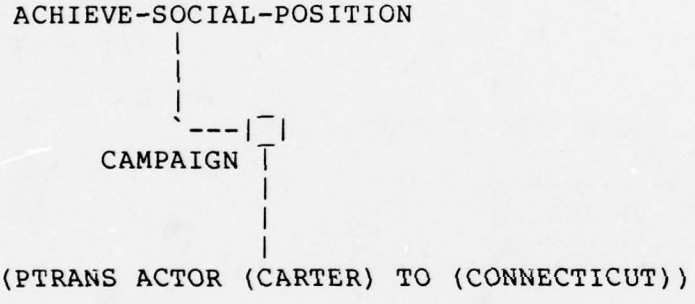
Presumably we as understanders have a memory entry for President Carter, and in fact has information about Carter which tells us some of the plans which he might be operating under, regardless of what action he actually performs. This information is encoded in ARTHUR under the PLANNER attribute for the memory entry for Carter. Some of the plans under this attribute include the CAMPAIGN memory structure (containing actions like shaking hands and kissing babies), any of the PERSUADE plans, and perhaps more. All of these plans, and any others listed under the PLANNER attribute for Carter, are added to the bottom-up index for the action of Carter PTRANSing to Connecticut. This means that ARTHUR

infers that any of Carter's actions may turn out to be part of a known plan of Carter's.

Consider the following example:

[40] President Carter wanted to improve his standing on the east coast. He went to Connecticut.

The first sentence gives Carter's implicit goal of being politically popular, which has a known plan of campaigning. ARTHUR can interpret his action of going to Connecticut in terms of that goal, since it generates a bottom-up inference of CAMPAIGN from any action Carter performs. This action would not seem to be part of a plan of campaigning if it were not performed by an actor who is known to have that plan. The explanation graph for this example is as follows:



[Figure 14]

Consider the following:

[41] "A snake was in the garden."

This is simply a LOCATION state conceptualization, with no action mentioned:

(LOCATION ACTOR (SNAKE) LOC (GARDEN))

If we as understanders have a memory entry for snakes as actors, we may infer a typical action or plan of a snake, in spite of the fact that we are not told any action the snake has performed. Similarly, when ARTHUR understands this sentence, the PLANNER attribute under the memory entry for SNAKE will be looked up. This attribute contains plans like CHANGE-HEALTH (NEG), meaning that a snake might hurt

somebody. These plans are added to the bottom-up index for the stated event of the snake being in the garden. Compare the following two examples:

[42] A snake was in the garden. Little Jimmy got bitten.

[43] A flower was in the garden. Little Jimmy got bitten.

In the case of [42], ARTHUR infers a CHANGE-HEALTH index from the first sentence, from the PLANNER attribute of the actor SNAKE. Without that inference, the fact that Jimmy got bitten would not make sense, as is the case in the second example [43].

Following is some more abridged ARTHUR output, to give an idea of how these indices are actually implemented and used in ARTHUR:

```
----------------
*** INPUT EVENT:
----------------
(LOCATION (ACTOR SNAKE) (LOC GARDEN))

---------------------------
GENERATING BOTTOM-UP INDEX:
---------------------------
((CHANGE-HEALTH (PLANNER SNAKE) (OBJECT NIL)))

-------------------------------
GENERATING DEFAULT EXPLANATION:
-------------------------------
(HEALTH (VAL -10) (PLANNER SNAKE) (OBJECT NIL))

-----------------------------------
GENERATING TOP-DOWN PREDICTIVE INDEX:
-----------------------------------
((CHANGE-HEALTH (PLANNER SNAKE) (OBJECT NIL)))
```

```
----------------
*** INPUT EVENT:
----------------
(PROPEL (ACTOR NIL) (OBJECT TEETH) (TO JIMMY))

--------------------------
GENERATING BOTTOM-UP INDEX:
--------------------------
((CHANGE-HEALTH (PLANNER NIL) (OBJECT JIMMY)))

PERFORMING SPECIFICATION SEARCH

------------------------
FOUND EXPLANATION-INDEX:
------------------------
((CHANGE-HEALTH (PLANNER SNAKE) (OBJECT JIMMY)))

UPDATING EXPLANATION GRAPH
```

3.2.3 Combining ACTOR, OBJECT and TO slots

Consider the following:

[44] President Carter went to the racetrack.

This is represented as follows:

```
(PTRANS ACTOR (CARTER) OBJECT (CARTER) TO (RACETRACK))
```

We have seen that the action of PTRANSing TO a racetrack accesses the LOCATION-SCRIPT attribute of the memory entry for racetrack, generating a bottom-up index of SEE(HORSERACE) and PLAY-GAME(WAGER). Similarly, since President Carter is the ACTOR of the action, the indices under the PLANNER attribute of ARTHUR's memory entry for Carter are added to the bottom-up index from the action as well. We said that these could be CAMPAIGN and any member of the PERSUADE package. Hence the complete bottom-up index from this action would be as follows:

```
(PTRANS ACTOR (CARTER) OBJECT (CARTER) TO (RACETRACK))
BOTTOM-UP INDICES:
------------------

SEE(HORSERACE)
PLAY-GAME(WAGER)
CAMPAIGN
PERSUADE
```

Since the PTRANS action might be explained through any of these indices, the action could turn out to be in service of Carter's goal of getting re-elected or of simply betting on a horse for fun. These indices therefore allow flexibility in arriving at an explanation for the action. The ARTHUR program can understand any of the following versions of this example:

> President Carter wanted to see the horserace. He went to the racetrack.
>
> President Carter wanted to win some money. He went to the racetrack.
>
> President Carter wanted to win some voter support. He went to the racetrack.
>
> President Carter wanted to meet with some potential campaign contributors. He went to the racetrack.

Each of these examples corresponds to a different one of the four bottom-up indices generated from the action of Carter going to the racetrack: SEE(HORSERACE), PLAY-GAME(WAGER), CAMPAIGN, PERSUADE.

Once more we present some abridged ARTHUR output to illustrate the implemented version of the above discussion.

```
---------------
*** INPUT GOAL:
---------------
(E-ENTERTAIN (PLANNER CARTER) (EVENT HORSERACE))

-------------------------------------
GENERATING TOP-DOWN PREDICTIVE INDEX:
-------------------------------------
((SEE (PLANNER CARTER) (EVENT HORSERACE)))

----------------
*** INPUT EVENT:
----------------
```

```
(PTRANS (ACTOR CARTER) (OBJECT CARTER) (TO RACETRACK))



---------------------------
GENERATING BOTTOM-UP INDEX:
---------------------------
((SEE (PLANNER CARTER) (EVENT HORSERACE))
 (PLAY-GAME (PLANNER CARTER) (COMPETITION WAGER))
 (CAMPAIGN (PLANNER CARTER))
 (ASK (PLANNER CARTER) (OBJECT NIL))
 (INFORM-REASON (PLANNER CARTER) (OBJECT NIL))
   .
   .      <abridged: all the persuade package plans
   .                 are inferred here>
   .  )

PERFORMING SPECIFICATION SEARCH

------------------------
FOUND EXPLANATION-INDEX:
------------------------
((SEE (PLANNER CARTER) (EVENT HORSERACE)))

UPDATING EXPLANATION GRAPH
```

3.2.4 Precondition indices

Recall the following example from the previous chapter:

[45] Mary wanted something to throw in a long and curving arc. She picked up her tennis racquet.

Understanding this story requires the reader to infer that Mary picked up her tennis racquet in order to throw it in a long and curving arc. Chapter 2 showed how non-standard uses of objects like this can be recognized when they occur in a story, without having been explicitly inferred from the action of getting the object. This is done by inferring a precondition index from certain actions, which are "flexibly" matched against top-down indices from a goal, if specification search fails to yield an explanation index.

Here we present the three explicit precondition rules for generating precondition indices from actions:

Rule 26: Control precondition

If an actor ATRANSes possession of a functional object to himself,
or an actor GRASPs a functional object,

Then infer that the action may satisfy a DCONT precondition for the performance of the object's function.

Rule 27: Proximity precondition

If an actor PTRANSes himself to the proximity of a functional object or to a location with an associated script,

Then infer that the action may satisfy a DPROX precondition for the performance of the object's function or the location's associated script.

Rule 28: Knowledge precondition

If an actor is the RECIPIENT of an MTRANS, MBUILD or ATTEND conveying the location of, or information concerning a functional object or a location with an associated script,

Then infer that the action may satisfy a DKNOW precondition for the performance of the object's function or the location's associated script.

On the basis of these rules, we make the following definition for convenient reference:

Definition: Precondition action

A Precondition action is any ATRANS, GRASP, PTRANS, MTRANS, MBUILD or ATTEND action which satisfies one of the above three Precondition Rules.

3.3 Top-Down Indices

Following is a list of 15 goals identified by Schank and Abelson [1977] and Wilensky [1978]:

SATISFY-HUNGER
SATISFY-SEX
SATISFY-SLEEP

ENJOY-ENTERTAINMENT
ENJOY-EATING
ENJOY-REST

ACHIEVE-ABILITY
ACHIEVE-POSSESSIONS
ACHIEVE-SOCIAL-POSITION

PRESERVE-HEALTH
PRESERVE-POSSESSIONS

DELTA-PROX
DELTA-KNOW
DELTA-CONTROL
DELTA-SOCIAL-CONTROL

The first 11 of these, i.e., all but the "DELTA-" goals, are all "theme-level goals". This means that performing a single bottom-up inference from any one of these goals yields a theme, or, in other words, the explanation for any one of these goals is a theme. According to the semantics of goal explanations as described in Wilensky [1978], this means that once one of these goals has been inferred, no further explanation is needed, other than to say that the goal arises from a known theme. Hence these goals have no bottom-up indices, i.e., they cannot be subgoals in service of any plan. The DELTA- goals, on the other hand, can serve as preconditions for other goals. The previous section contained a discussion of actions that could be inferred to satisfy DELTA-goal preconditions of plans for goals. ARTHUR only uses DELTA-goals to represent precondition plans for theme-level goals.

All of the 15 goals have top-down indices, that is, inferring top-down from any of the goals can give rise to plans which are in service of those goals. Some of these plans are scripts, like $GROCER, and some are named plans, like FIND. The top-down indices which are used in this thesis for these goals are given in the following table.

TABLE 1
Top-Down Goal Indices

SATISFY-HUNGER	FIND(FOOD), $RESTAURANT, $GROCER, EAT
SATISFY-SEX	FIND(SEXOBJECT) + PERSUADE
SATISFY-SLEEP	GOTOBED, $HOTEL
ENJOY-ENTERTAINMENT	SEE(SHOW), READ, PLAY-GAME, INTOXICATION
ENJOY-EATING	$RESTAURANT, $COOKOUT, $GROCER, EAT
ENJOY-SEX	FIND(SEXOBJECT) + PERSUADE
ENJOY-REST	$HOTEL, FIND(BED)
ACHIEVE-ABILITY	EXERCISE, SCHOOL
ACHIEVE-POSSESSIONS	PERSUADE PACKAGE
ACHIEVE-SOCIAL-POSITION	PERSUADE PACKAGE
PRESERVE-HEALTH	COMPLY(THREAT), AVOID
PRESERVE-POSSESSIONS	COMPLY(THREAT), AVOID
DELTA-PROX	USE-VEHICLE, WALK
DELTA-KNOW	PERSUADE PACKAGE, READ
DELTA-CONTROL	PERSUADE PACKAGE, COMPLY
DELTA-SOCIAL-CONTROL	POLICE, PERSUADE, COMPLY

In the previous chapter, we saw the top-down index for SATISFY-HUNGER, and an example of its use. Note that unlike actions, these theme-level goals contain usually only a single slot: the ACTOR who has the goal. For example, the representation of Willa's being hungry is as follows:

(SATISFY-HUNGER ACTOR (WILLA))

Hence the goal itself, and not the slotfillers, are the source of information contributing to the top-down index from an instantiated goal. In fact, consider the following sentence:

[46] Jimmy Carter was hungry.

If we know some specific plans that Carter has for satisfying his hunger, then we could infer them from this statement. However, by and large, special characters still use the same methods as anyone else to achieve their goals. ARTHUR does not recognize any special plans on the part of certain actors that might override the standard plans for achieving a goal. Hence the top-down indices for a given

goal will always be the same each time a particular goal is read by ARTHUR.

A detailed analysis of the top-down indices given in the above table is unnecessary. We have seen one example of the use of a top-down index from a goal, SATISFY-HUNGER. Many more such top-down indices will be exampled in subsequent chapters. Hence we will not at this stage examine these top-down indices any more closely, but rather we will let them get presented naturally, as they arise one by one in examples.

3.4 Summary

This chapter has presented the rules by which ARTHUR generates bottom-up indices from actions and top-down indices from goals. We have seen that the bottom-up index from an action contains the same information as would be contained in the bottom-up inferences generated by a step-by-step understander from the same action. These bottom-up indices depend on the action and all slotfillers in an instantiated action, just as the step-by-step inferences from an action depend on the information from those slotfillers. In contrast, the top-down indices from a goal depend only on the goal itself, since theme-level goals do not contain slots other than the ACTOR who has the goal.

3.5 End of one chapter, beginning of the next

The intersection of top-down and bottom-up indices yields an explanation index for an action-goal pair. The explanation index uniquely specifies the inferential path through memory that must be followed to get from the action to the goal. Hence we can represent such an explanation path by an explanation graph, which indicates only the action, explanation index and goal, thereby uniquely denoting the explanation path, from which a fully expanded explanation path could be reconstructed.

This method of arriving at explanation indices for action-goal pairs is achieved when there exists an intersection between the top-down and bottom-up indices. Such an intersection will not always be non-null. When specification search fails to yield an explanation index, the reader must resort to changing some previous inference in order to explain the new input. Chapter 4 presents the process of supplanting explanations, which occurs in precisely the cases where there is a null intersection between the bottom-up and top-down indices corresponding to an action - goal pair.

CHAPTER 4

SUPPLANTING INFERENCES

4.1 Introduction

The previous two chapters presented a process whereby connections can be found between goals and actions, by indexing the top-down and bottom-up inferences from those goals and actions. However, story events do not always conform to our previous inferences; stories can be misleading, causing readers to change their minds about their own previous inferences. People are able to understand story events that they in no sense predicted ahead of time. This chapter presents a process by which an understander can notice contradictory input and change its mind about its own previous inferences in order to account for the new input. *This process* is implemented in the ARTHUR program.

A story can contain an event which contradicts some natural inference we may have made from a previous event. Consider the following:

[47] Mary picked up a magazine.

Q1) Why did Mary pick up a magazine?
A1) Probably to read it.

[48] Mary picked up a magazine. She swatted a fly.

Q1) Why did Mary pick up a magazine?
A2) To use it to swat a fly.

Chapter 1 mentioned that these were deceptively simple stories. This section will examine these examples in more detail. To see the complexities of understanding these examples, consider what is required to answer questions about them. First of all, answering question Q1 requires the reader to have understood the intentions underlying

Mary's actions; i.e., finding an explanation for her actions in terms of her goals. Answer A1 reflects the reader's inference that Mary probably wanted to perform the default known function of a magazine: reading it. Answer A2 indicates that the reader inferred that Mary wanted to use another, less common function of a magazine: using it as a flyswatter.

Notice that the first sentence of [48] is identical to sentence [47]. Hence the reader would give answer A1 after the first sentence of [48], even though he gives answer A2 after the second sentence. This means that the reader has made two different inferences about the intention underlying Mary's action of picking up the magazine. The problem is that the original interpretation of Mary's picking up the magazine is still perfectly valid. She may have initially picked it up to read it, and then decided to swat a fly with it. Or, she may have picked it up to read it and then swatted a fly with some other object, not related to the magazine at all. The following is a summary of the three possible interpretations of [48]:

(48a) Mary picked up a magazine to read it. She then was annoyed by a fly, and she swatted it with the magazine she was holding.

(48b) Mary picked up a magazine to read it. She then was annoyed by a fly, and she swatted it with a flyswatter that was handy.

(48c) Mary picked up a magazine to swat a fly with it.

Interpretation (48c) is the one that corresponds to the reader's answer A2 above, and most readers agree with the interpretation. However, it is difficult to point to anything in example [48] that indicates that this interpretation is preferable to the others. None of the three answers can be faulted on purely logical grounds, and as far as explicit story statements go, there is no reason to assume that swatting the fly was connected to picking up a magazine. Yet readers strongly prefer that connection, even at the expense of their initial inference that Mary might read the magazine.

There are many possible inferences we could make about why Mary picked up the magazine. The default function of a magazine is to be read, but there are myriad other functions, like tacking up the cover on the wall, or jotting down a telephone number in a margin. A reader can understand a story which imputes a totally unfamiliar use to a magazine, as in the following:

[49] Mary picked up a magazine. She wanted something to gnaw on.

Q1) Why did Mary pick up a magazine?
A2) To gnaw on.

Again, the reader's answer reflects an inference that Mary's second action not only used the magazine as the object of the gnawing, but also that her reason for picking up the magazine was using it for this purpose. There are three possible interpretations of [49], just as there were for [48]:

(49a) Mary picked up a magazine to read it. She then wanted something to gnaw on, so she gnawed on the magazine she was holding.

(49b) Mary picked up a magazine to read it. She then wanted something to gnaw on, so she got a piece of bread out of the breadbox and gnawed on that.

(49c) Mary picked up a magazine to gnaw on.

Using a magazine to gnaw on is such an unlikely function that it would never be inferred spontaneously by a reader. Yet in [49], as in [48], the reader prefers that new inference over the initial default inference that Mary had planned to read the magazine.

Hence the reader prefers to infer a highly unusual function of a magazine over its default function in both stories [48] and [49]. These inferences are preferred even though the original default inference could provide a consistent interpretation of the stories. The problem is to determine what information and what processes are involved in arriving at interpretations (48c) and (49c) for stories [48] and [49].

4.2 Supplanting inferences

The reason interpretations (48c) and (49c) are preferred over the other possible interpretations of the above stories is that they are the most parsimonious explanations of the stories. If we assume that Mary's intention was the same throughout the story, then the resulting explanation of the story contains only one goal, which accounts for both of Mary's actions. In [48], that goal is swatting a fly, and in [49] the goal is gnawing on something.

To understand stories [48] and [49], the reader must recognize that the initial inference (that Mary wanted to read the magazine) cannot serve to explain both of Mary's actions. Furthermore, the reader must discover another goal (swatting or gnawing) which can account for both of her actions. Finally, the reader must decide to infer that this new goal was Mary's actual intention, and must change its mind about its initial inference.

Changing your mind about an inference on the basis of subsequent input in a story is called supplanting the inference. Understanding [48] and [49] requires the reader to supplant the initial inference that Mary might read the magazine, by the new inference that she might swat a fly with it or gnaw on it, respectively. Arriving at this interpretation requires the reader to perform three processes:

1 - Recognize that an initial inference fails to explain a new input

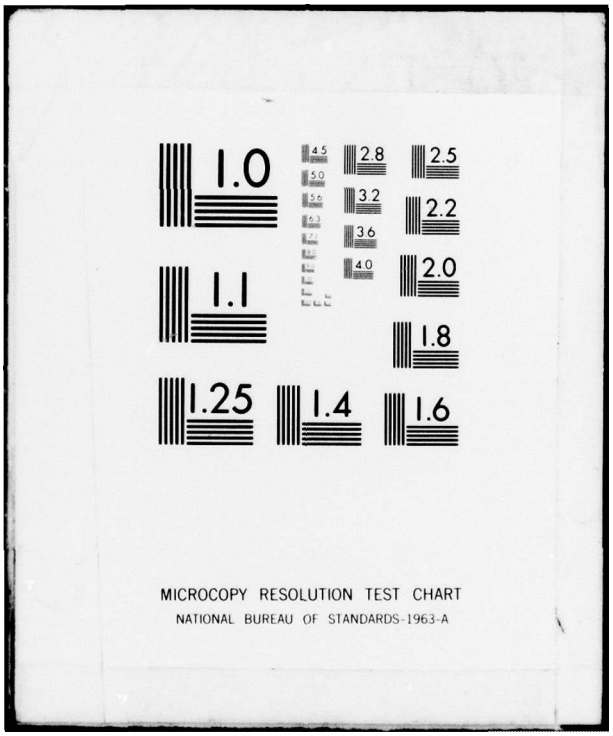
2 - Generate an alternative to that inference which can parsimoniously explain both old and new inputs.

3 - Supplant the old inference by the new one in the representation of the story.

The ability to supplant inferences is important in story understanding because it allows the reader to correctly interpret a story even when the story is misleading. For example, in [48] and [49], the story first implies that Mary might want to read a magazine, but then Mary uses the magazine for an entirely different purpose. Understanding this story requires the reader to realize that the initial implication has been overridden by the subsequent implication. The reader must then update its explanation of the story to reflect the fact that Mary's intention is now inferred to be swatting a fly, and that reading the magazine is no longer considered to have been her intention. That is, the correct explanation for Mary's

AD-A081 012 YALE UNIV NEW HAVEN CONN DEPT OF COMPUTER SCIENCE F/G 5/7
ADAPTIVE UNDERSTANDING: CORRECTING ERRONEOUS INFERENCES.(U)
JAN 80 R H GRANGER N00014-75-C-1111
UNCLASSIFIED RR-171 NL

2 OF 3
AD
A081012


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

action depends on the reader's ability to supplant inferences.

4.3 Overview of the process

We have said that people have the ability to understand stories that contain apparent contradictions by changing their minds about inferences which are contradicted by subsequent input. This ability is implemented in ARTHUR in the process of supplanting inferences. There are three basic phases in the process of understanding a story involving a contradiction:

1 - A new story input is read and interpreted;

2 - A contradiction is noticed between the new input and a previous inference;

3 - The contradiction is resolved by supplanting the contradicted inference by a new inference.

In the following sections we will outline the processing that takes place during each of these phases.

4.3.1 Interpreting a new input

Chapters 2 and 3 presented the rules by which ARTHUR interprets new inputs. We will briefly restate the interpretation process here. Recall that ARTHUR differentiates between two different kinds of input: actions and states on the one hand and goals on the other hand. The process of interpretation differs according to which of these two broad classifications the story input falls into. In this section we will concentrate on inputs which are actions. Later we will see what happens when a new goal is either read or inferred by the ARTHUR understander.

INTERPRETING AN ACTION OR STATE INPUT

STEP 1: GENERATE A BOTTOM-UP INDEX

This is the process of generating a set of pointers to the possible bottom-up inference paths from an action. The bottom-up index specifies the possible plans and scripts that this action might be part of, and the causal

consequences that the action might have.

STEP 2: FIND KNOWN GOALS

Find any goals which might serve as the explanation for this action: i.e., any goals which this action might be interpreted as being in service of. If there are no known goals, generate a default goal to explain the action.

STEP 3: SPECIFICATION SEARCH

Intersect the top-down and bottom-up indices from the goal and action, respectively. The resulting intersecton set specified the possible inferential paths from the action to the goal.

4.3.2 Noticing a contradiction

Chapter 2 showed that ARTHUR's way of finding a connection between a new action and an existing goal is by performing specification search on the indices from action and goal. In that chapter, we only saw examples in which specification search successfully yielded an explanation index, thus specifying a connection between the action and goal. In this section, we will see that the cases in which specification search fails to yield an explanation index are precisely the cases in which the action input contradicts a previous inference.

4.3.2.1 Contradicting a goal inference

Consider the following:

[50] John went to the gas station. He robbed the station and got away with $50.

When the first event is read, that of John going to the gas station, there have not yet been any goals inferred for John. Hence ARTHUR infers a default goal to explain John's action: the action was probably part of the $GAS-STATION script, which is the typical set of actions associated with going to a gas station and buying gas. The default goal inferred for this action is ACHIEVE-POSSESSION of gasoline, i.e., ARTHUR infers that John was simply going to the gas

station to buy gas in the standard "scripty" manner.

When ARTHUR reads the second event, of John robbing the gas station, it has already inferred that John wanted to buy gasoline. Now ARTHUR attempts to interpret John's new action as being in service of that inferred goal. This is done by performing specification search on the indices from goal and action. The top-down index from that goal is the $GAS-STATION script. The bottom-up index from the action of robbing the station is the set of PERSUADE package plans THREATEN, OVERPOWER and STEAL. (The word "rob" includes information specifying both an action (ATRANS) and a plan (THREATEN, OVERPOWER, STEAL). The representation of "robbery" is an ATRANS plus a pre-specified bottom-up index including these three plan indices. Chapter 3 discusses this issue, and gives some other examples of words which specify both an action and a bottom-up plan index.)

There are no elements in common between these top-down and bottom-up indices. Hence specification search will fail: the intersection of the two indices is the null set. This means that ARTHUR cannot find any inferential path to connect John's robbery action with his inferred goal of buying gas at the station. we call this situation SPECIFICATION FAILURE.

Recall that the result of specification search is an index to an inferential path connecting a given goal to a given action. A successful specification search means that the action has been interpreted as having been performed in service of the goal.

Conversely, specification failure between a given goal and action means that the action was not performed in service of the goal.

Rule 30: Specification failure

If specification search between the indices from an action and a goal yield the null set,

Then infer that the action was not performed in service of that goal.

For instance, in the above example, John's action of robbing the gas station was not in service of his goal of buying gasoline. As we will see, the reason for that is that John probably never actually had that goal of buying gasoline at all. ARTHUR inferred that goal on the basis of John's action of going to the gas station, but there are of course other possible reasons for someone to go to a gas station, even though the most likely reason is to get gas.

4.3.2.2 Contradicting a plan inference

Sometimes we will infer the correct goal for a character, but we will incorrectly infer the plan he is going to use to satisfy the goal. For example, consider the following:

> [51] Carl was bored. He picked up the newspaper. He reached under it to get the tennis racket that the newspaper had been covering.

The first sentence states that Carl's goal was to entertain himself somehow. From the second sentence, we are most likely to infer that Carl is planning to read the newspaper to assuage his boredom. If we are asked after reading the second sentence why Carl picked up the newspaper, we will most likely answer that he probably wanted to read it because he was bored. Then the third sentence says that he wanted to get his tennis racket. That contradicts our inference that he picked up the newspaper to read it. Instead, we now infer that he was planning to play or practice tennis, and that he only picked up the newspaper to enable him to get to the tennis racket. The action of picking up the newspaper is now considered to be a precondition to Carl's real plan of getting his tennis racket.

In either case Carl's action was performed in service of the goal of entertaining himself, but the plan that we infer he is performing has changed. When ARTHUR makes the initial plan inference that Carl is going to read the paper, that inference will become the constrained top-down index from the goal. That is, from then on ARTHUR will be expecting Carl to fulfil his entertainment goal by reading the paper. Hence ARTHUR will again notice the contradiction because of a specification failure: the bottom-up inferences from Carl's action of getting the tennis racket will fail to intersect with the top-down "read" inference from the entertainment goal.

4.3.3 A contradiction leads to a disconnected explanation

When a story event fails to be explained by any previous inferences, it means that we cannot find any connection between that event and the rest of the story. Hence we can end up with separate explanations for each separate event in a story. We call this situation a DISCONNECTED EXPLANATION. We will see that a disconnected explanation is a reflection of a lack of understanding of the story.

When specification failure occurs, ARTHUR is left with an action for which it has no goal explanation. This is also the case when ARTHUR reads the first action in a story: no goal has yet been inferred, so there is no goal to explain the action. Recall that ARTHUR's reaction to this situation is to generate a default goal from the action, and that newly inferred goal then serves as the explanation for the action. Rule 29 from Chapter 2 states this rule of when to infer a default goal:

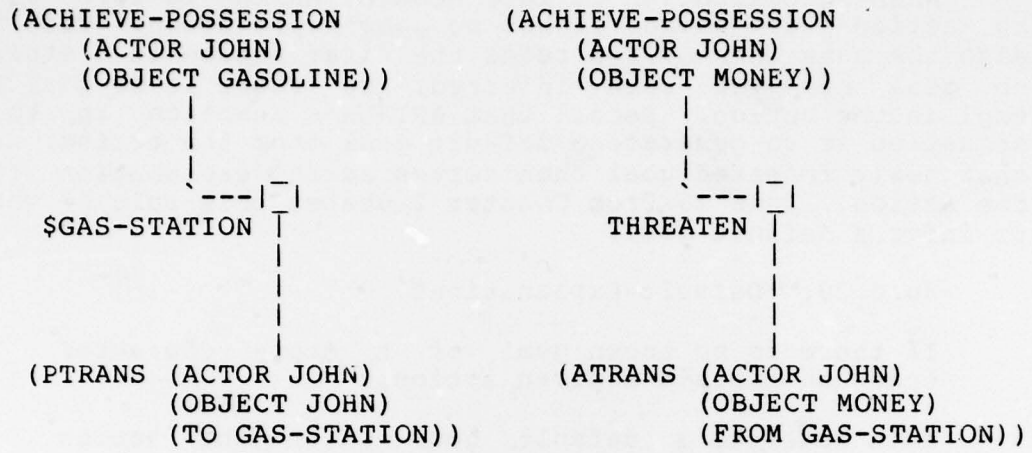
> Rule 29: Default Explanations
>
> If there is no known goal of a story character that can explain a given action,
>
> then generate a default goal which the action might be in service of. That goal is then considered to be the current explanation of the action.

When this rule was introduced in Chapter 2, the only situation to which it applied was when ARTHUR was reading the first action in a story, and hence it had not yet inferred any goals at all. However, recall example [50]:

> [50] John went to the gas station. He robbed the station and got away with $50.

In this example, ARTHUR inferred a goal for John, but John's action of robbing the station failed to be interpreted in terms of that goal. Hence Rule 29 applies in this situation too, and ARTHUR infers a default goal for the action. The default goal for the robbery action is ACHIEVE-POSSESSION of money.

ARTHUR now has read two actions in this example, and has generated two separate default goals, one for each of them. The current explanation graph for the example is as follows:



[Figure 15]

This graph means that ARTHUR has inferred that John went to to the gas station in order to get gasoline, and he robbed the gas station in order to get money. ARTHUR so far knows of no connection between these two actions and explanations.

This is an example of a disconnected explanation: there are two actions and two separate explanations, and no connection between them. We will see that this reflects a lack of understanding of the story.

4.3.4 What's wrong with disccnnected explanations

We saw that the above "robbery" example led us to a disconnected explanation: the two events in the story were explained by two separate goal explanations with no connection between them.

This explanation is incorrect: that is, it represents a lack of understanding of the example. We can see this best by examining the question-answering behavior that this graph implies. If ARTHUR were asked "Why did John go to the gas station", on the basis of this explanation graph its answer would be that John wanted to buy gasoline. We recognize this to be incorrect at this point in the story: we may have imputed that goal to John on the basis of his first action, but the second action leads us to change our minds about that initial goal inference.

As we shall see, whenever an explanation graph contains two or more separate explanations which are not connected by any inferential path, the explanation graph does not reflect complete understanding of the story. This leads us to the following two rules:

Rule 32: Connected explanations

If there are two or more goals in a story representation,

then every goal must be connected to at least one other goal in the representation.

Rule 33: Disconnected explanations

If an understander arrives at a disconnected explanation for a story,

then the understander has not yet understood the story.

These rules are based on properties of both stories and human story understanders. Stories are usually written in such a way that unconnected goals are not presented: every new goal is related in some way to previous goals. Human understanders assume that any story they read will conform to this property. Hence even if we read a new goal that cannot be related in any known way to some other known goal, we will tend to try to make further inferences to make such a connection. For example, consider the following example due to Schank [1975]:

[52] John wanted to be chairman of the department. He got some arsenic.

If asked why John got some arsenic, understanders will usually answer that he may have wanted to use it to kill the current chairman of the department, so that John could take over. This is an admittedly extreme example of people's strong tendency and ability to infer connections between actions and goals.

We can imagine a story which might seem to contain disconnected explanations at first, but a connection becomes apparent after more of the story is read. For example, consider the following:

[53] John loved to fix washing machines. He was on his way to his mother's house. On the way he saw a tulip. ....

This story has mentioned a goal and two actions, but they don't all seem related in any clear way. It might be the case that John is going to his mother's house to fix her washing machine, and people's answers to questions about this story usually reflect this connective inference. However, few people assume that John's seeing a tulip has anything to do with his goal of fixing washing machines. Rather, we may assume that John might pick the tulip and bring it to his mother, if he does anything at all with it. Story characters can of course have multiple goals, and a goal like wanting to bring your mother a present can appear in a story without necessarily being connected to other events in the story.

If the story ended with John's seeing the tulip, it seems very abrupt, and we might very well feel that we didn't understand it. Specifically, we wonder why the tulip was mentioned at all and what it has to do with the rest of the story. This corresponds to our rule that says that understanding depends on having a connected explanation.

If, however, the story goes on to say that John did something with the tulip, like bring it to his mother, then we can interpret his seeing it as having been a precondition for whatever he then did with it. Hence at that point we will have connected his seeing the tulip with one of his goals, and thus we will no longer have a disconnected explanation. In the meantime, any interim assumption we might make about what John might do with the tulip will serve as a tentative explanation, since we have the ability to supplant that explanation later. Hence an understander's ability to supplant explanations allows the understander to make tentative connecting inferences for all story inputs, since such inferences can be changed later if subsequent input suggests a better explanation.

4.3.5 Resolving a contradiction

The previous sections have presented rules which a reader can use to recognize when a story input contradicts one of its inferences. This recognition was tied to the occurrence of a specification failure during understanding. Understanding a story which contains such a contradiction requires the reader to come up with a new inference to account for the contradictory input. This section will outline the processes involved in coming up with new inferences to resolve an apparent contradiction.

When a new story input contradicts a previous goal inference, there are two possible reasons for that contradiction: either the story character changed his mind about achieving his goal, or else the understander inferred the wrong goal for the story character. In this section we

will outline these two situations.

4.3.5.1 A story character changes his mind

Consider the following example, taken from Wilensky [1978]:

> [54] John wanted to watch the football game. He also had a paper due the next day.

Wilensky [1978] uses this example as an illustration of a case of GOAL CONFLICT: the character John has two goals, and the understander can infer that due to a time limitation, John will only be able to achieve one of the goals, at the expense of the other. In this example, we first infer one goal for John and then another one: John wants to watch the football game according to the first sentence and John wants to write a paper according to the second sentence. Having generated these two inferences, we must then find some connection between them, or else we have not understood the story.

Wilensky [1978] describes GOAL CONFLICT, GOAL COMPETITION and GOAL CONCORD as the three basic relationships that can exist between two or more goals. We adopt Wilensky's rules for determining these goal relationships, and apply them here. This thesis will present some examples where these goal relationships occur, but we will not deal at any length with these issues; we simply adopt Wilensky's methods.

In example [54], the two goals are found to be in conflict with each other. Finding one of the the three goal relationships constitutes finding a connection between the goals, thereby satisfying our rule that we must have a path between the goals in a story representation.

Rule 34: Goal relationships

If an understander has generated a disconnected explanation,

then check to see if the goals in the explanation satisfy any known goal relationship.

4.3.5.2 The understander must change his mind

Consider the following:

[55] John got a gun with a silencer. He wanted to use it as part of his 'hit man' costume for the halloween party.

From the first sentence of [55], we are likely to assume that John is going to use the gun for its primary function, i.e., shooting. Since the gun has a silencer, we will probably infer that John wants to shoot a person, since that is the most likely reason for wanting to be surreptitious about it. If we are asked, after the first sentence, "Why did John get a gun", an appropriate answer is that he probably wanted to use it to kill someone. This answer reflects the fact that we have inferred John's probable goal.

We may also have inferred many other possible reasons John may have had for getting the gun, even though we typically only mention one when answering a question. However, we hypothesize that the goal of wanting the gun as part of a costume is not among those that we infer from the first sentence. The question of whether we generate this specific inference is moot: the point is that there is certainly SOME inference that we didn't make from that action, and yet we can understand a story in which that unpredicted inference turns out to be John's actual goal.

In [55], we first infer a set of natural inferences from the action of getting a gun with a silencer, and then we read the second sentence. This sentence states that John's actual reason for getting the gun was to use it as part of a "hit man" costume. This goal has no apparent connection with the previously inferred goal of killing someone. Hence we are led to a disconnected explanation: we have inferred two separate goals in the story with no connection between them.

According to Rule 34 as described in the previous section, we should now check to see whether any of the three goal relationships described by Wilensky [1978] are applicable to this disconnected explanation. If we apply Wilensky's rules for determining the presence of such relationships, we will find that none of them is applicable: for example, it is not the case that John has both of these goals, and that the goals are in conflict with each other. John actually has only one of these goals, that of using the gun for his costume. It is we, the understanders, who have spuriously inferred the other goal, that he might want to use the gun to shoot someone.

Rule 35: Undoing a goal inference

If a disconnected explanation cannot be connected by any known goal relationships,

then assume that the earlier of the two disconnected goal inferences was inferred erroneously.

It may be the case that the goal inference is not at fault, but only the plan that we inferred to be in service of that goal. In such a case, we only undo the plan inference.

Rule 36: Undoing a plan inference

If a disconnected explanation cannot be connected by any known goal relationships,

and the initial goal is known to be correct,

then assume that the plan inferred in service of that goal is erroneous.

When a goal has been inferred erroneously, the solution is to correct the error: i.e., undo the inference. However, we cannot simply negate the first inference and leave it at that, because we still need an explanation for John's first action of getting the gun. What ARTHUR does is to hypothesize that the new goal of using the gun as part of a costume is the explanation for that action. ARTHUR then needs a way to test this hypothesis. Recall that ARTHUR's normal understanding process involves testing every new event in a story to see if it can be explained in terms of existing goal inferences. Hence ARTHUR can test this new hypothesis the same way: by performing specification search between the top-down index from the new goal and the bottom-up index from the action of getting the gun.

(Observe that this is just what would have happened if the story had been reversed: we would have read about John's goal of having a gun for his costume first, and then we would have used the predictions from that goal to interpret why John got a gun. In that case, no erroneous inferences would have been generated, since the top-down predictions from the goal would have allowed us to choose the correct inference from the action immediately.)

By performing specification search on these two indices, John's action will be found to be a precondition for his goal, and hence the action will have been explained by the goal. Now we say that the new goal has SUPPLANTED the initially inferred goal as the explanation for this action. Thus ARTHUR has arrived at a connected explanation, by supplanting the initial "kill" explanation with the new "costume" explanation.

Rule 37: Supplanting a goal inference

If a goal inference in a disconnected pair has been assumed to have been inferred erroneously,

then supplant that inference by the correct inference, as the explanation for the action that gave rise to the inference.

Recall that a plan inference might be in error instead of a goal inference. In such a case, we said that we infer that the plan was erroneous but we leave the goal inference intact. Hence we then attempt to explain the new action in terms of the original goal, but without constraining the top-down index from that goal. We then must find a new connection to the goal from the action that gave rise to the erroneous plan.

Rule 38: Supplanting a plan inference

If a plan inference has been inferred erroneously,

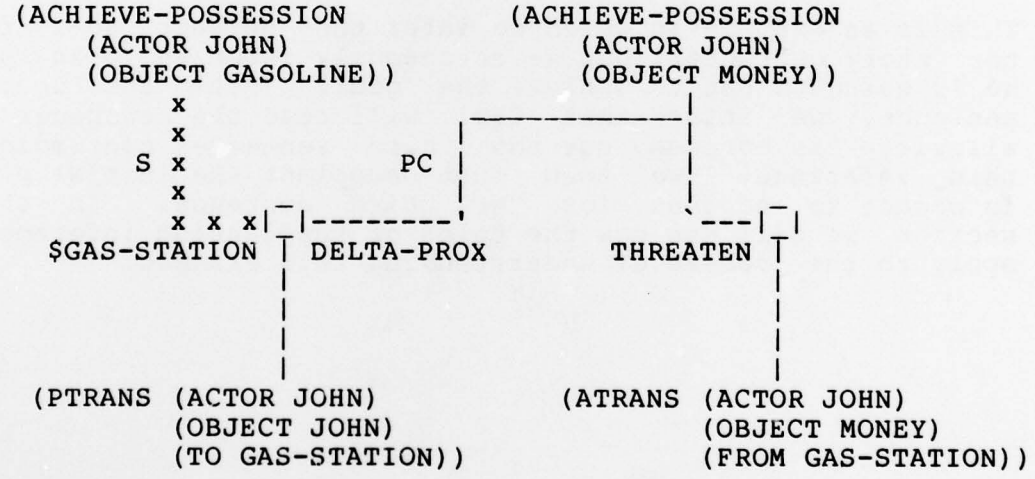
then generate an unconstrained top-down index from the original goal, and re-explain any actions in terms of the newly inferred plan for that goal.

4.3.6 Representing supplanted explanations

Recall example [50]:

[50] John went to the gas station. He robbed the station and got away with $50.

The previous sections have shown how the reader arrives at a connected explanation for this example, by supplanting the initial inference about John's goal. The resulting explanation can be represented by an explanation graph, as in the previous chapters. The explanation graph for the explanation of [50] is as follows:



[Figure 16]

This graph contains a new link: the link from the PTRANS action up to the original ACHIEVE-POSSESSION(GASOLINE) goal is marked with an "S", meaning that this explanation for the action has been supplanted. The new explanation for the PTRANS action is represented by the precondition link

(labelled "PC") linking the action to the ACHIEVE-POSSESSION(MONEY) goal. (Precondition links were described in Chapter 2.)

The graph means that the understander initially thought that John went to the gas station so that he could get gas, but now the understander thinks that John went to the gas station so that he could rob it. John robbed the station because he wanted money.

4.3.7 Supplanting a plan inference

The previous sections have presented rules for recognizing a contradiction in a story and resolving the contradiction by supplanting a goal inference. Recall that a story may contradict a plan inference without contradicting any goal inferences. In this section the rules presented above will be applied to the situation of recognizing and supplanting an erroneous plan inference.

Recall the following example:

[51] Carl was bored. He picked up the newspaper. He reached under it to get the tennis racket that the newspaper had been covering.

This is an example in which we infer the correct goal for the story character, but we erroneously infer the plan that he is going to use to achieve the goal. From the second sentence, we infer that Carl will read the newspaper to alleviate his boredom, but the third sentence contradicts this inference. We then must supplant the initial plan inference to account for the third sentence. In this section we will see how the rules of supplanting inferences apply to the process of understanding this example.

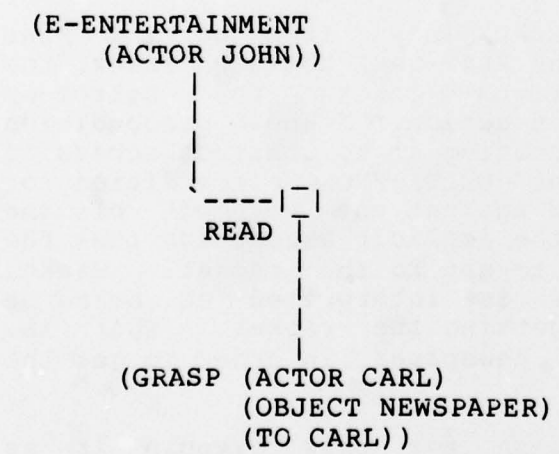
4.3.7.1 Interpreting new input

The first sentence of [51] contains an implicit goal statement: if someone is bored, then he has the goal of entertaining himself. This goal becomes part of the story representation directly, as in the rules in Chapter 2. Once we have generated this goal, we generate a top-down index from it, corresponding to the plans that we predict Carl will use to achieve his goal. This top-down index is generated according to the rules in Chapter 3. The index consists of the plans SEE(EVENT), READ, PLAY-GAME, INTOXICATION.

The second sentence says that Carl picked up a newspaper. A newspaper is a functional object. ARTHUR has an entry under the FUNCTION attribute of NEWSPAPER in its memory, giving a set of bottom-up indices corresponding to the known functional uses of a newspaper. One of those bottom-up indices is READ. ARTHUR performs specification search on the top-down index from the goal and the bottom-up index from the action. The intersection is the plan READ. Hence Carl's action of picking up the newspaper is interpreted as being part of a plan to read the newspaper, in service of his goal of entertaining himself.

This inference causes ARTHUR to constrain the top-down index from the entertainment goal to just include READ and its preconditions. This is because ARTHUR no longer predicts that Carl will do any of the four possible plans for entertainment listed above. Now ARTHUR believes that Carl will do the READ plan to satisfy his goal.

The explanation graph at this point is as follows:



[Figure 17]

4.3.7.2 Noticing and supplanting a contradiction

The final sentence of example [51] says that Carl picked up a tennis racket that the newspaper had been covering. The bottom-up index from this action is computed from the functionality of a tennis racket, just as the bottom-up index from getting the newspaper was generated. That index includes the PLAY-GAME(TENNIS) plan.

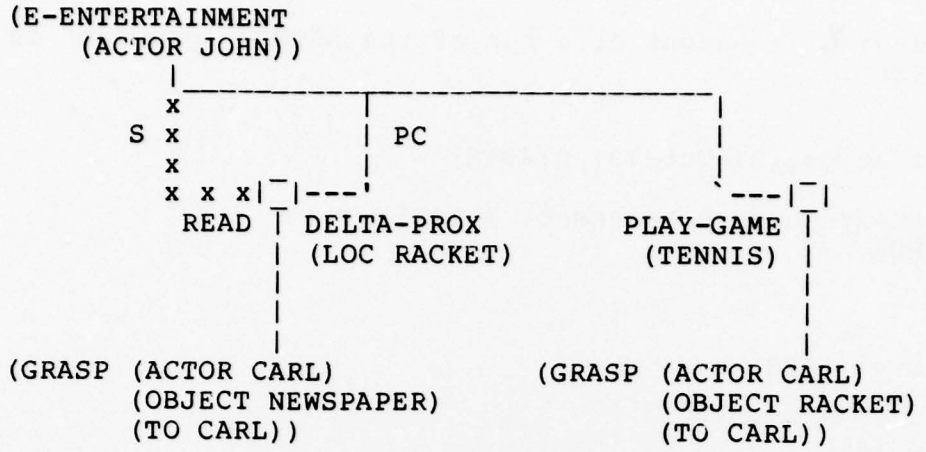
ARTHUR performs specification search on the constrained top-down index from the goal and the bottom-up index from this action. Since ARTHUR inferred that Carl was going to do the READ plan to satisfy his entertainment goal, the top-down index from that goal has been constrained to consist of just the READ plan and its preconditions DKNOW DPROX and DCONT. There are no elements in common between this index and the index from the action of getting the racket. This means we have a specification failure.

We now have no explanation for the action of getting the racket. However, in this example we know that Carl's goal of entertaining himself was correctly inferred, because it was stated implicitly in the first sentence of the example. Rule 36 says that if we know a goal to be correct, then asssume that the plan in service of the goal was inferred erroneously. Hence we infer that the READ plan is not how Carl will satisfy his goal.

By Rule 38, we then re-interpret the action of picking up the racket in terms of the entertainment goal. This time, the specification search yields a plan index of PLAY-GAME(TENNIS) instead of READ. Now by the same rule, we must re-interpret the action of picking up the newspaper in terms of this new PLAY-GAME plan.

Since the bottom-up READ index from getting the newspaper fails to match the PLAY-GAME top-down index, the specification search algorithm checks the bottom-up precondition indices from that action. A known precondition function of any object can be using it to obstruct access to another object. Hence, the DELTA-PROX precondition for getting the racket is matched against the movement of the newspaper, thereby making the implicit assumption that the newspaper was moved in order to get to the racket. Hence, picking up the newspaper is interpreted as being a DELTA-PROX precondition for getting the racket. That is, Carl had to pick up the newspaper in order to get the racket.

The final explanation graph for this example is as follows:



[Figure 18]

This graph means that the understander initially thought that Carl's action of picking up the newspaper was part of a plan to read the newspaper to entertain himself. Now the understander believes that the action was performed as a precondition, so that Carl could get to a racket, as part of a plan of playing tennis in order to entertain himself.

4.3.7.3 ARTHUR output

Following is output of a run of the ARTHUR program on example [50].

```
[PH: Initiation. 31-Oct-79  9:49AM]

 Yale TOPS-20 Command Processor 3A(415)-2
@RUN ARTHUR

*(ARTHUR)

Input story: *EX8


The story is:

 JOHN WENT TO THE GAS STATION.
 JOHN ROBBED THE GAS STATION.

((PTRANS (ACTOR JOHN)
         (OBJECT JOHN)
         (TO GAS-STATION))

 (ATRANS (ACTOR JOHN)
         (OBJECT MONEY)
         (TO JOHN)
         (FROM GAS-STATION)
         (BUI (THREATEN OVERPOWER STEAL))))

***  ARTHUR: UNDERSTANDING PHASE ***
```

ARTHUR OUTPUT	ANNOTATION

NEXT SENTENCE:	

JOHN WENT TO THE GAS STATION.	
*** INPUT EVENT:	The first input is a PTRANS
(PTRANS (ACTOR JOHN)	event.
(OBJECT JOHN)	
(TO GAS-STATION))	
GENERATING BOTTOM-UP INDEX:	The bottom-up index is
(($GAS-STATION (ACTOR JOHN)))	generated in this case
FOR EVENT:	from the FUNCTION att-
(PTRANS (ACTOR JOHN)	ribute of the TO slot,
(OBJECT JOHN)	GAS-STATION. That
(TO GAS-STATION))	location has a known
	script index associated
GENERATING DEFAULT EXPLANATION:	with it, $GAS-STATION.
(A-POSSESS (ACTOR JOHN)	The default explanation
(OBJECT GASOLINE))	is that John did the
	gas station script
NEW EXPLANATION	because he wanted to
(A-POSSESS (ACTOR JOHN)	get gasoline.
(OBJECT GASOLINE))	This fact is added to
ADDED TO EXPLANATION-GRAPH	the story representa-
	tion.
UPDATING EXPLANATION-GRAPH:	
EXPLANATION TRIPLE:	The explanation graph
GOAL: (A-POSSESS	now represents the
(ACTOR JOHN)	PTRANS action as being
(OBJECT GASOLINE))	explained by the goal
EVENT:	of getting gas, via
(PTRANS	the gas-station script.
(ACTOR JOHN)	
(OBJECT JOHN)	
(TO GAS-STATION))	
INDEX:	
(($GAS-STATION	
(ACTOR JOHN)))	
STATUS: ACTIVE	

```
--------------
NEXT SENTENCE:
--------------
 JOHN ROBBED THE GAS STATION.

*** INPUT EVENT:
(ATRANS (ACTOR JOHN)
        (OBJECT MONEY)
        (TO JOHN)
        (FROM GAS-STATION)
        (BUI (THREATEN
              OVERPOWER
              STEAL)))

GENERATING BOTTOM-UP INDEX:
((THREATEN (ACTOR JOHN))
 (OVERPOWER (ACTOR JOHN))
 (STEAL (ACTOR JOHN)))
FOR EVENT:
(ATRANS (ACTOR JOHN)
        (OBJECT MONEY)
        (TO JOHN)
        (FROM GAS-STATION)
        (BUI (THREATEN
              OVERPOWER
              STEAL)))

PERFORMING SPECIFICATION SEARCH

*** SPECIFICATION FAILURE ***

GENERATING
DEFAULT EXPLANATION:
(A-POSSESS (ACTOR JOHN)
           (OBJECT MONEY))
FOR INDEX
((THREATEN (ACTOR JOHN))
 (OVERPOWER (ACTOR JOHN))
 (STEAL (ACTOR JOHN)))

NEW EXPLANATION
(A-POSSESS (ACTOR JOHN)
           (OBJECT MONEY))
ADDED TO EXPLANATION-GRAPH

UPDATING EXPLANATION-GRAPH:
```

The second sentence is an ATRANS event. This event specifies its own bottom-up index, via the "BUI" slot in the CD, as was described in Chapter 3. The BUI slot constrains the possible bottom-up indices that can be generated from this action.

The three specified indices are instantiated and generated as the bottom-up index.

ARTHUR attempts to explain the new event in terms of the previously inferred A-POSSESS (GAS) goal. This attempt fails.

ARTHUR then generates a default goal from the new action's index, which is not yet connected with the rest of the explanation graph.

```
EXPLANATION TRIPLE:
  GOAL: (A-POSSESS
         (ACTOR JOHN)
         (OBJECT GASOLINE))
     EVENT:
        (PTRANS
         (ACTOR JOHN)
         (OBJECT JOHN)
         (TO GAS-STATION))
     INDEX:
        (($GAS-STATION
          (ACTOR JOHN)))
     STATUS: ACTIVE
EXPLANATION TRIPLE:
  GOAL: (A-POSSESS
         (ACTOR JOHN)
         (OBJECT MONEY))
     EVENT:
        (ATRANS
         (ACTOR JOHN)
         (OBJECT MONEY)
         (TO JOHN)
         (FROM GAS-STATION)
         (BUI (THREATEN
               OVERPOWER
               STEAL)))
     INDEX:
        ((THREATEN
          (ACTOR JOHN)))
     STATUS: ACTIVE

*** SUPPLANTING EXPLANATION

PERFORMING SPECIFICATION SEARCH
```

ARTHUR now has arrived at a disconnected explanation for the story so far. The next step will be to try to connect it by checking for goal relationships and then supplanting the initial goal inference.

No goal relationships are found, so ARTHUR attempts to supplant the initial inference by the new inference.

```
(PTRANS (ACTOR JOHN)
        (OBJECT JOHN)
        (TO GAS-STATION))
IS PRECONDITION FOR
(A-POSSESS (ACTOR JOHN)
           (OBJECT MONEY))

(A-POSSESS (ACTOR JOHN)
           (OBJECT GASOLINE))
SUPPLANTED BY
(A-POSSESS (ACTOR JOHN)
           (OBJECT MONEY))
AS EXPLANATION FOR EVENT
(PTRANS (ACTOR JOHN)
        (OBJECT JOHN)
        (TO GAS-STATION))
VIA EXPLANATION INDEX
((DELTA-PROX
        (ACTOR JOHN)
        (TO GAS-STATION)))
```

Specification search between the new goal and the previous action is successful, yielding a precondition link between going to the gas station and robbing the gas station. This match is found because PTRANS is a precondition action (see Chapter 3), and the location "gas station" plays a role in the subsequent THREATEN index. Hence, the initial inference is successfully supplanted by the new inference, as the explanation for John's going to the gas station in the first place.

```
*** FINAL EXPLANATION-GRAPH:

EXPLANATION TRIPLE:
  GOAL: (A-POSSESS
         (ACTOR JOHN)
         (OBJECT GASOLINE))
     EVENT:
        (PTRANS
         (ACTOR JOHN)
         (OBJECT JOHN)
         (TO GAS-STATION))
     INDEX:
        (($GAS-STATION
         (ACTOR JOHN)))
     STATUS:
        (SUPPLANTED
         (A-POSSESS
          (ACTOR JOHN)
          (OBJECT MONEY)))
EXPLANATION TRIPLE:
  GOAL: (A-POSSESS
         (ACTOR JOHN)
         (OBJECT MONEY))
     EVENT:
        (ATRANS
         (ACTOR JOHN)
         (OBJECT MONEY)
         (TO JOHN)
         (FROM GAS-STATION)
         (BUI (THREATEN
               OVERPOWER
               STEAL)))
```

The final explanation graph specifies two goals for the PTRANS action, the first of which is marked as having been supplanted by the second.

```
                                    |
     INDEX:                         |
        ((THREATEN                  |
          (ACTOR JOHN)))            |
     STATUS: ACTIVE                 |
                                    |
     EVENT:                         |
        (PTRANS                     |
         (ACTOR JOHN)               |
         (OBJECT JOHN)              |
         (TO GAS-STATION))          |
     INDEX:                         |
        ((DELTA-PROX                |
          (ACTOR JOHN)              |
          (TO GAS-STATION)))        |
     STATUS: ACTIVE                 |
                                    |
                                    |
*** PROCESSING COMPLETED            |
*** READY FOR QUESTIONS             |
                                    |
```

```
Input question:*EX8Q1


QUESTION:
 WHY DID JOHN GO TO THE GAS STATION?

ANSWER:
 AT FIRST I THOUGHT IT WAS BECAUSE HE WANTED TO GET SOME
GASOLINE, BUT ACTUALLY IT'S BECAUSE HE WANTED TO GO THERE
TO STEAL MONEY.


Input question:*EX8Q2


QUESTION:
 WHY DID JOHN ROB THE GAS STATION?

ANSWER:
 BECAUSE HE WANTED MONEY.


Input question:*#

[PH: Termination. 31-Oct-79  9:51AM. PS:<GRANGER.TH>G1.LOG.1]
```

4.4 Summary

We have seen that a story event can contradict inferences that an understander has previously made. Either goal or plan inferences can be contradicted. In order to understand a story which contains such a contradiction, we must be able to change our minds about the contradicted inferences. We may have to come up with new inferences to account for the otherwise unexplained contradictory event.

ARTHUR recognizes a contradiction by the occurrence of a specification failure, i.e., the failure to explain a new event in terms of any known goal. ARTHUR resolves contradictions by the process of supplanting a goal or plan inference. Supplanting an inference involves undoing an erroneous inference and generating a new inference. The new supplanting inference must be able to explain the new contradictory event and it must be able to substitute as the new explanation for whatever events gave rise to the original erroneous inference.

CHAPTER 5

FORCING AN INTERPRETATION

5.1 Introduction

Stories sometimes contain wholly novel information. A story can imply or explicitly state a new inference rule, which then must be used in understanding the rest of the story. Understanding such a story requires the reader to first recognize the new inference rule, and then incorporate it into memory in such a way that it will be properly applied in understanding the story so as to give rise to previously unknown inferences from story statements.

This chapter presents categories of new inference rules, and processes by which a reader can recognize and then use such rules while understanding a story. This ability enables the reader to understand stories which require knowledge beyond the inference rules which are built into its memory.

Consider the following stories:

[56] John was hungry. He went outside and collected some ants.

Q1) Why did John get some ants?
A1) He may have wanted to eat them.

[57] John was hungry. He went outside and collected some ants. John was the "human anteater" at the circus.

Q2) Why did John get some ants?
A2) He wanted to eat them.

Understanding story [56] and generating answer A1 requires the reader to infer that John might plan to eat ants to satisfy his hunger. Ants are not known to function as food, so the inference that John might eat them violates the reader's inference rule that says that people only eat food to satisfy their hunger. Hence understanding this story requires the reader to infer a novel connection between the action of getting ants and the goal of satisfying hunger.

Story [57] contains an explicit statement saying that John regularly eats ants. Hence generating answer A2 requires the reader to use this statement about John's eating habits to infer a novel connection between his action of getting ants and his goal of satisfying his hunger.

Both of these stories lead the reader to generate a previously unknown fact about the world: that a particular actor may eat ants. This fact contains information about an object (ants), and about an actor's plan (eating) in service of a goal (satisfying hunger).

Facts about objects and about ways of satisfying goals are encoded in ARTHUR as inference rules: ARTHUR generates inferences from what it reads based upon facts such as these. Hence new facts can be encoded in ARTHUR in the form of new inference rules. For example, once the reader knows the above fact about John and ants, then whenever the story next says that John got some ants, the reader can infer that John may plan to eat the ants. This inference is not one that would have been made prior to the introduction of this new fact. The reader had no inference rule which would generate EATing as a bottom-up inference from the action of getting ants. Hence making this inference requires the reader to add a new inference rule to its own memory, such that that rule will give rise to this new inference.

There are two processes that a reader must perform to understand these stories:

1 - Recognize a new inference rule in a story, whether implied, as in [56], or explicitly stated, as in [57].

2 - Incorporate the new inference rule into memory such that the rule can be used to generate previously unavailable inferences during understanding of the story.

Depending on whether the rule is implicit or explicit in the story, the reader must address the following two problems:

1 - The reader must be able to recognize the need for a new inference rule in order to find a connection between otherwise disconnected story inputs.

2 - The reader must be able to recognize the existence of a new inference rule which is explicitly stated in a story.

Inferring an implied new inference rule requires the reader to generate the rule itself first, and then to add the rule to its memory to be used as a new inference rule in understanding the rest of the story. This chapter is primarily concerned with the *recognition* and use of story-supplied inference rules. Near the end of the chapter is a discussion of the process of recognizing and generating a new inference rule which is implied but not stated in a story.

5.1.1 Stories containing new inference rules

Consider the following stories:

[58] John's boss fired him. John went out and got a gun

Q1) Why did John get a gun?
A1) He may want to shoot his boss for revenge.

[59] John's boss fired him. People who lose their jobs often commit suicide. John went out and got a gun.

Q1) Why did John get a gun?
A3) He may want to shoot himself.

Giving answer A1 to question Q1 required the reader to have found a connection between the two events in [58]. To find such a connection, the reader must have made inferences from the events, since no connection was explicitly stated in the story. A natural inference from John getting a gun is that he might want to shoot someone. A plausible inference from John's boss firing him is that John may want revenge. Shooting someone is a known method of achieving revenge. Hence these inferences enable the reader to find a connection between the story events.

Giving answer A2 to question Q1 required the reader to have found a different connection between the same two events in example [59]. Answer A2 indicates that the inferences made from the events in [59] were different from those made in [58]. In this case, the inference from John's boss firing him was that John might want to be dead. Shooting oneself is a known plan for becoming dead, and getting a gun can be part of a plan of shooting yourself.

Since the two events are identical in both stories [58] and [59], the difference between the inferences the reader generated from these events must be due to the third sentence of [59]. That sentence by itself does not make us infer that someone might commit suicide. The sentence contains information which causes the reader to make different inferences from other statements in the story. The problem is to identify the information contained in that sentence, and to determine how it affects the inferences made from other story statements.

5.2 Story-supplied inference rules

The difference between [58] and [59] is that [59] contains a statement which provides a novel connection between the events in the story. The inferred connection between John's being fired and his getting a gun in [59] is that being fired led John to plan suicide, which the gun is instrumental to. That connection is novel in that the reader did not generate that connection in reading story [58]. The connection is provided by the addition of some new information: being unemployed can lead to planning suicide. When combined with the knowledge that a gun can be used for suicide, this new information provides the reader with a novel connection between the two events in the story.

To understand example [59], the reader must realize that a new relation has been asserted between the state of unemployment and the plan of committing suicide. Furthermore, the reader must apply that relation to this particular instance, in order to infer that since John has become unemployed he may plan to commit suicide. Once we have made that inference, based on the relation asserted in [59], we can then recognize that John's action of getting a gun may be part of his plan to commit suicide. Understanding example [59] required us to use the relation in the third sentence as a new inference rule. An inference rule tells us what inference to make from a given input in a story. Similarly, the relation in [59] told us what inference to make from the state of being unemployed.

A statement that specifies the inferences to make from a given story input is a statement of a new inference rule. We call such a statement a STORY-SUPPLIED INFERENCE RULE. Story-supplied inference rules are a general story device which enable a story to make a novel connection between otherwise unconnected story inputs. An inference rule can be supplied by a story whenever the events in the story are related in unusual ways, especially if the relation might be so unusual as to be unknown to the reader. A story-supplied inference rule can enable the understander to make the appropriate inferences required to connect up the events in a story.

A story understanding system must be able to recognize and use the inference rules supplied by a story in order to find the appropriate connections between statements in the story. For example, consider the following:

[60] John was feeling sick. He went out and drank some Jameson's. He often drinks Jameson's because it's good for his small intestine.

This story supplies a new inference rule in the final sentence: drinking Jameson's can be part of a plan for improving an intestine's health. Feeling sick implies that the victim will want to be healthy. Getting healthy can be achieved by a plan for improving one's health. Drinking Jameson's is not part of any known plan for improving health. However, the story-supplied inference rule in [60] states that drinking Jameson's can be part of a plan for improving the health of the small intestine. Improving the health of part of your body makes you healthier. Thus an understander can find a connection between John's illness and his drinking only by using the story-supplied inference rule to infer that John's drinking can lead to improving his health.

A story-supplied inference rule contains the following components:

1. A story input, which can be an action, state or goal. In example [60], the input was the action of drinking Jameson.

2. An inference from that story input, whose category (goal, plan, action, state) depends on the category of the input. In example [60], the inference was that the action could be part of a plan of changing one's health.

The ability to understand story-supplied inference rules is important in story understanding because it allows the reader to find novel connections among the elements of a story, thereby finding explanations for those story elements. For example, in [60], a reader would need to infer that the reason John drank Jameson's was to improve his health. To make this inference, the reader needs to understand that the story-supplied inference rule tells us to make this inference from the action of drinking Jameson's. That is, the explanation for why John drank Jameson's depends on the information contained in the story-supplied inference rule.

5.3 Recognizing story-supplied inference rules

To understand a story that supplies a new inference rule, the reader must first be able to recognize that a rule has been stated. Recognizing new inference rules requires the reader to check every explicitly mentioned relationship between two story inputs, to see if the relationship is already known or not.

There are two basic classes of inference relationships that can occur in story-supplied inference rules: top-down inferences from a goal and bottom-up inferences from an event or state. The bottom-up inferences from an event can either be novel plans that this particular event can be part of, or new causal consequences that the event can give rise to. The top-down inferences from a goal can either be novel plans for achieving that goal, or novel causal state changes that can either help or hinder the achievement of the goal.

I found it necessary to further distinguish three sub-types of inference information about novel plans from an action. They are: novel functions of objects, plans associated with locations and idiosyncratic reasons for performing actions. Altogether, ARTHUR can recognize and distinguish among the following six types of inference information:

1. A novel function of an object
2. A plan associated with a location
3. Idiosyncratic reasons for actions
4. An unusual causal consequence of an event
5. Idiosyncratic methods of achieving a goal
6. A causal state change helping or hindering a goal

This section describes these six types of inference information that can be provided by story-supplied inference rules, and outlines the knowledge that the reader can use to recognize each type of information.

5.3.1 A novel function of an object

Information about a novel function of an object is a type of inference information. Presenting a novel function of an object can cause a reader to make different inferences from actions which involve that object. For example, if a story says that a screwdriver can be used to start a car, then getting a screwdriver can be inferred to be part of a plan to start a car. Chapter 3 discussed the rules for making inferences from functional objects.

A story-supplied inference rule can provide inference information about a novel function of an object. For example, consider the following:

[61] John was hungry. He picked a tulip. He knew that tulips are edible and nutritious.

When someone is hungry, they want to satisfy their hunger. Picking a tulip is not part of any known plan for satisfying hunger. However, the story-supplied inference rule in the final sentence of this example contains the information that tulips are edible. This information provides the reader with a previously unknown function of a tulip: being eaten.

Rule 39: A novel function of an object

If a relation is asserted between an object and a plan,
and the plan is not a known function of the object,

then the relation is a new inference rule specifying a plan associated with the object

Once the reader has inferred this new function of a tulip, he can then infer that the action of INGESTing a tulip can be part of a plan to eat food, which can be in service of a goal of satisfying hunger. Since getting a tulip is a precondition for using some function of the object, John's action of picking a tulip can be interpreted as being in service of satisfying his hunger.

5.3.2 Plans associated with a location

Knowledge about a location at which an event takes place can sometimes give rise to plan inferences. For example, if John goes to the swimming pool, we infer that he is going to swim there. Chapter 3 gives the rules that govern how an understander uses location information to generate inferences.

A story-supplied inference rule can provide information about a typical plan or script associated with a location. Such information can cause the reader to make novel inferences from being at a certain location. For example, consider the following:

[62] John went to the railroad station. The flea market sale took place by the tracks every week. John was looking for a good antique lamp.

Understanding this story requires the reader to understand the inference rule supplied in the second sentence. This inference rule contains information about a typical activity that is associated with the location of the railroad tracks. Understanding this information causes the reader to infer that John may have gone to the train station in order to browse in the flea market. This inference allows the reader to find a connection between John's action of going to the station and his goal of acquiring an antique lamp. Without the story-supplied inference rule, the reader would not be able to connect the action with the goal.

Rule 40: A plan associated with a location

If a relation is asserted between a location and a plan,
and the plan is not a known location-index of the location,

then the relation is a new inference rule specifying a plan associated with the location.

5.3.3 Idiosyncratic reasons for an action

An actor can perform an action as part of an idiosyncratic plan or script. To understand a story about a particular actor's actions, the reader needs to know what idiosyncratic reasons that actor may have for performing the action. Idiosyncratic plans can include the use of functional objects or "scripty" locations. For example, if John typically goes into the bathroom to practice the oboe, then the plan of improving oboe playing is a valid inference when John goes into the bathroom, but it is not necessarily a valid inference when anyone else does so.

Story-supplied inference rules can contain information about an actor's idiosyncratic reasons for performing certain actions. For example, consider the following:

[63] Jonn bought an Andrew Wyeth original. He likes using original canvasses to serve fancy dinners on. His new girlfriend Jane was coming for dinner.

To understand [63], the reader must recognize that the inference rule in the second sentence tells us an idiosyncratic plan that John has. The information provided about this plan causes the reader to infer that John will use the Wyeth to serve dinner to Jane.

Rule 41: An idiosyncratic reason for an action

If a relation is asserted between an actor's action and a plan of the actor,
and the action is not a known part of the plan,

then the relation is a new inference rule specifying that the action can be a part of the plan when performed by this actor.

In the above example, a story-supplied inference rule informed the reader that John liked to acquire original paintings to serve dinner on. This is an example of a story character who has added an additional plan to the existing repertoire of possible reasons for performing a particular action. Conversely, story characters sometimes intentionally limit themselves to performing certain actions only for certain specified purposes. A story-supplied inference rule can contain information restricting the possible reasons why an actor might perform a certain action. For example, consider the following two stories:

[64] Yesterday John needed to get some groceries. First he went out and bought a 357 magnum. Then he went to the grocery store.

[65] John is a strictly non-violent pacifist who only uses guns to give demonstrations at his weekly anti-gun meetings. Yesterday John needed to get some groceries. First he went out and bought a 357 magnum. Then he went to the grocery store.

Understanding example [64] will most likely lead the reader to infer that John might plan to steal groceries from the store, since that is the most natural inference that can connect the two story events of getting a gun and going to a store. (This inference may turn out to be supplanted later, but it is by far the most natural inference from this example.) In contrast, understanding [65] requires the reader to recognize that John will not use a gun for any of its typical functions, such as the plans of threatening or overpowering people. Hence the reader will not infer that John will rob the store, but rather that he may simply buy some groceries. The most plausible inference from John's action of buying a 357 magnum in this story is that he may

use it in a demonstration at an upcoming meeting.

The difference between these two examples is that [65] contains an inference rule which restricts the possible plans to be inferred from possession of the functional object "gun".

> Rule 42: Restricting the possible reasons for an action
>
> If a particular plan is said to be an actor's only reason for performing a certain action,
> and there exist other known plans that this action could be part of,
>
> then the relation is a new inference rule constraining the allowable plans inferred from the actor's action.

Understanding this inference rule requires the reader to restrict the inferences made from the event of John getting a gun. We will see that a restricting inference rule like this is implemented by constraining the bottom-up inferences from an action. Constraining inferences is discussed in Chapters 2 and 4.

5.3.4 Unexpected causal consequences of events

Events can have causal consequences, and those consequences can affect actor's goals. Chapter 3 gave the rules by which a reader can infer the effects of causal consequences on goals. A story-supplied inference rule can specify a previously unknown causal effect of an event. For example, consider the following:

> [66] Running too fast always gives Mary a temperature. She ran for the bus yesterday morning, and now she's in the hospital.

In order to understand this story, the reader must make the causal inference provided by the inference rule in the first sentence. The rule says that running has the causal consequence of changing Mary's health.

Rule 43: Unexpected causal consequences of events

If a relation is asserted between an event and a causal consequence,

and the event is not known to cause that consequence,

then the relation is a new inference rule specifying that the event can give rise to the causal consequence.

Once the reader has understood this inference rule, making the "change-health" inference from Mary's running for the bus allows the reader to connect that action with the fact that she had to go to the hospital. Going to the hospital can be a plan in service of the goal of preserving health; and running, in Mary's case, can give rise to a change in Mary's health, which can induce a goal of preserving health.

5.3.5 Idiosyncratic plans for achieving a goal

An actor can have his own personal and idiosyncratic methods for achieving goals. A story-supplied inference rule can specify an actor's idiosyncratic method of attaining a particular goal. In order to understand a story in which an actor makes use of such a method, the reader must be aware of the idiosyncrasy. For example, consider the following:

[67] John was hungry. He didn't eat a thing when Jane took him to MacDonald's. John only eats macrobiotic.

The final sentence of [67] is an inference rule. In states that John has an idiosyncratic method of satisfying his hunger: eating only macrobiotic food. This is an example of an actor limiting himself to only certain possible plans for achieving a particular goal.

Rule 44: Idiosyncratic methods of achieving goals

If a particular plan is said to be an actor's only method of achieving a given goal,
and the goal is known to be attainable by other plans,

then the relation is a new inference rule constraining the allowable plans that this actor can perform to achieve this goal.

To understand why John didn't eat at MacDonald's when he was hungry, the reader must understand the idiosyncratic plan specified by the inference rule in the final sentence of [67]. This rule says that when John has the goal of SATISFY-HUNGER, he can achieve that goal only by performing the plan of EATing food which has the feature of being MACROBIOTIC.

An actor can also have a novel method of attaining a goal which is in addition to the standard plans for this goal. For example, consider the following:

[68] John was hungry. He filled up a bowl with ants. He's the 'human anteater' in the circus.

Understanding this story requires the reader to make a connection between John's goal of satsifying hunger and his action of filling up a bowl with ants. Making this connection requires the reader to understand the inference rule supplied in the third sentence, which says that John can eat ants.

Rule 45: Novel methods of achieving a goal

If a plan is asserted to be in service of a particular actor's goal,
and the plan is not known to be in service of that goal,

then the relation is a new inference rule specifying that this actor can achieve this goal by performing this plan. specifying that the goal can be achieved by the actor performing this plan.

5.3.6 Helping and hindering a goal

A causal result of an event can affect an actor's ability to achieve a goal. The effect can be either positive (helping) or negative (hindering). Chapter 3 describes the rules by which a reader can determine the effects of causal consequences on goals and plans. A story-supplied inference rule can specify a relationship between a causal consequence of an event and a plan. For example, consider the following:

> [69] John wanted to rake the whole yard clean by sundown. When the grass gets wet, raking becomes almost impossible. John saw storm clouds gathering in the sky. He decided the job wouldn't get done.

Understanding this story requires the reader to find a connection between the action of storm clouds appearing and John giving up his goal. The key to finding this connection is given in the story-supplied inference rule in the second sentence.

> Rule 46: Helping and hindering a goal
>
> If a causal state change is asserted to either help or hinder a particular actor's goal,
> and the state change has no known relation to that goal,
>
> then the relation is a new inference rule specifying that the actor's goal may be helped or hindered by the causal state change.

This rule says that when something causes the grass to increase its moisture state, that state change hinders the plan of raking. Storm clouds are non-intentional actors that can cause rain, which increases moisture. Since John plans to clear the yard by raking it, the causal result of the storm clouds raining on the yard will hinder his plan to rake it. Hence this plan will fail to achieve the goal of clearing the yard. The helping and hindering relations between goals and state changes are described in Chapter 3.

5.4 Understanding stories using new inference rules

Understanding a story that contains a new inference rule can require the reader to use that new inference rule in order to find a connection between the events in the story. For instance, example [69] above contains a rule that says that moisture can hinder raking. To understand the implied connection between the arrival of storm clouds and John's abandonment of his goal in that story, the reader nees to use the information in the new inference rule along with knowledge that storm clouds can cause increased moisture.

To use a new inference rule to understand a story, the reader must first recognize when a story input is affected by the new rule. One possible way for a reader to recognize the applicability of a rule to an input would be to check every new rule against every new input. This approach suggests a possible combinatorial explosion, if there are many inference rules and many story inputs, especially since the rule must also be checked against all possible inferences from each input. If there are only one or two new rules in a relatively short story, then this approach would be feasible.

The problem of recognizing the applicability of a new inference rule is the same as the problem of recognizing the applicability of any inference rule, including those that are built in to the understander's memory. Chapters 2 and 3 described how an understander can use an indexed storage scheme for inferences to enable an understander to generate bottom-up or top-down inferences from an event or goal directly. This is possible because the appropriate inference rules are pre-stored as attributes of actions, actors, objects, locations and goals in the understander's memory.

The next section will outline the knowledge necessary for an understander to integrate new story-supplied inference rules into its memory, in accordance with the indexing and storage schemes presented in Chapters 2 and 3. Then the following section will present a set of rules which a reader can use to understand a story using built-in inference rules together with new story-supplied inference rules which have been integrated into memory.

5.4.1 Integrating a new inference rule into memory

Integrating a new inference rule into memory requires the understander to know precisely what information to store and where to store it. Recall that a new inference rule consists of a story input and an inference to be made from that input. For instance, example [61] in this chapter contained an inference rule which specified that the plan of eating food could be inferred from the action of picking a tulip. Integrating this inference rule into memory means storing the EAT FOOD plan inference under the GRASP TULIP action.

The previous section identified six different types of information that could be specified in a new inference rule. They were as follows:

1. A novel function of an object
2. Plans associated with a location
3. Idiosyncratic reasons for actions
4. Unexpected causal consequences of events
5. Idiosyncratic plans for achieving a goal
6. Aiding and hindering a goal

Each of these categories contains information about a different aspect of the input. For example, a rule specifying a novel function of an object contains information pertaining to that object, and an idiosyncratic plan for achieving a goal contains information pertaining to the actor.

The point of integrating a new rule into memory is to cause the specified inference to be generated whenever the specified input occurs in the story. The indexed understanding scheme described in the previous chapters will cause such inferences to be generated if they are stored under the appropriate attributes of the memory entries for the specified inputs. For example, integrating a new rule about a novel function of an object into memory requires that the understander store this new function under the FUNCTION attribute of the object. Doing so will cause the new inference specified in the rule to be generated when this object's FUNCTION attribute is referenced, which will happen whenever the object occurs as the OBJECT of certain actions. (Chapter 3 outlines the rules governing references to attributes of objects.)

Thus the determination of where to store a new inference rule depends on the type of input specified in the rule. The following table specifies where a new rule is stored according to the category of the rule.

THE RULE SPECIFIES:	STORE UNDER:
a function of an object	the FUNCTION attribute of the object
an inference associated with a location	the LOCATION-INDICES attribute of the location
an actor's idiosyncratic reason for an action	the PLANNER attribute of the actor
an unexpected causal consequence of an event	the CAUSATION attribute of the event
an actor's idiosyncratic plan for achieving a goal	the PLANNER attribute of the actor
a causal state change's effect on a goal	the CAUSAL attribute of the goal

[Figure 19]

5.4.2 What an inference rule specifies

Recall the following example:

[66] Running too fast always gives Mary a temperature. She ran for the bus yesterday morning, and now she's in the hospital.

We said that the first sentence of this story contained an inference rule specifying an unexpected causal consequence of an event. Specifically, the event of PTRANSing in a rapid manner results in a negative health change. The above table says that this new rule should be stored under the CAUSATION attribute of the event that gives rise to it. In this case, under the BUI attribute of PTRANS would be a rule checking for each of the slotfillers of the instantiated action in the example. Hence the rule could be expressed as follows:

If a PTRANS action occurs with ACTOR MARY and MANNER RAPIDLY,

then infer that the action may lead to a CHANGE-HEALTH (NEG) of OBJECT MARY.

This is an example of index augmentation of an action: a new index is added to the list of any existing indices that may be inferred from the action. Index augmentation involves appending the new index as the first element under the specified attribute in memory. If there are other indices listed, they are still inferred, but the new index becomes the default.

Rule 47: Index augmentation

If a new attribute of some element of an action is specified by a new inference rule,

then that attribute is added as the first in the list of indices of that element.

Consider the following:

[70] Guns are edible.

This is a statement of a new fact about guns. Hence, it augments the FUNCTION index of the object GUN with the plan index EAT. There are already known functions of guns: THREATEN and OVERPOWER. Hence after index augmentation, the memory entry for gun is as follows:

```
GUN
---
 FUNCTION:
 ---------
 EAT
 THREATEN
 OVERPOWER
```

Now consider the following continuation to [70]:

[71] Guns are edible. Yesterday John bought a gun.

From this example, ARTHUR will infer that John may have gotten the gun because he wanted to eat it. This inference is made because we have not yet read any explicit statement of John's goals, and the default inference from getting a gun has been augmented to be eating the gun.

Now consider the following:

[72] Guns are edible. Yesterday John bought a gun. He had gotten some threatening phone calls.

This story will require ARTHUR to first infer that John might plan to eat the gun, and then to infer that he wanted the gun to protect himself, via a counter-threat to the threat he had received. This final inference is possible because the THREATEN and OVERPOWER inferences are still generated from gun, even though they are not the default inferences.

Recall the following:

[73] John was hungry. He didn't eat a thing when Jane took him to MacDonald's. John only eats macrobiotic.

The last sentence of this example specifies an actor's idiosyncratic plan for a goal, and is stored in the PLANNER attribute of that actor. In this case, John satisfies his hunger by eating only macrobiotic food. This is an example of index constraint: the plans inferred when John has an S-HUNGER goal are limited to only the specified plans. In this case, John will only perform the EAT, $RESTAURANT and $GROCER indices when the FOOD rolefiller has the attribute MACROBIOTIC. Index constraint causes the list of indices from an action or goal to be lessened.

Rule 48: Index constraint

If a new inference rule specifies that a particular attribute of some element of an action or goal is the only such attribute allowable,

then that attribute becomes the first and only element in the list of indices for that action or goal.

Index augmentation and index constraint are the two types of changes that a new inference rule can specify. Either way, the new rule becomes the default inference from the specified object, and in the case of augmentation the other inferences from that object are maintained, while in the case of constraint the other inferences are invalidated.

5.5 Making improbable assumptions

Consider the following example:

[74] John was hungry. He got some ants.

If asked why John got ants, an appropriate answer would be that he may have planned to eat them. Giving this answer requires the reader to have inferred a new fact which was not explicitly stated in the story. The inferred fact is that John might eat ants. This is a highly improbable assumption, since ants are not known to the reader to be edible. However, readers answers to questions indicate that they make this inference in the process of understanding this story.

The process of making improbable assumptions consists of three steps:

1 - Recognize the need for a new fact to allow understanding of the story

2 - Generate a new fact

3 - Treat the fact as a new inference rule.

Consider ARTHUR's processing of example [74]. The first sentence states an S-HUNGER goal for John. Then the second sentence states an action, which fails to be explained in terms of the S-HUNGER goal by specification search. ARTHUR next generates a default explanation for the action. In this case, ants have no known function, so the explanation is a loose-end DCONT ANTS index: that is, infer that John simply wants to have the ants for some as-yet-unknown reason.

This is an example of a circumstance in which ARTHUR cannot supplant an inference. There are two reasons for this. First of all, there is no previous action to re-explain: the S-HUNGER goal inference was explicitly stated in the story, not inferred from an action. Hence, the inference cannot have been inferred erroneously. We call an explicitly stated explanation, such as this S-HUNGER goal, an invulnerable explanation, since it cannot be supplanted. Secondly, in this instance there is no new goal inference which could supplant a previous one. Loose-end explanations are indices, not goal explanations, and therefore cannot serve as supplanting inferences. This can be summarized in the following two rules:

Rule 49: Invulnerable explanations

If a story character's goal was explicitly stated in the story, not inferred from a stated action,

then that goal cannot be supplanted by a subsequent goal inference, since the initial goal could not have been erroneously inferred.

Rule 50: Supplanting by loose end

If an explanation is a loose end,

then that explanation cannot supplant any previous explanation.

Hence we have a situation in which the reader has a disconnected explanation, but cannot supplant an inference to create a connected explanation. In this circumstance, the understander attempts to force an interpretation: in this case, to interpret the action of getting ants in terms of the S-HUNGER goal. Note that the understander could not have reached this point without having already tried to connect that action to that goal, via the first understanding step of specification search. Since that attempt failed, the understander now assumes the need for some new fact which could give rise to a new inference rule which would serve to connect the disconnected explanation explanation.

Rule 51: Recognition of need for new fact

If both specification search and supplanting fail to result in a connected explanation by the end of a story,

then a new fact must be inferred which can give rise to a connective inference rule.

Having recognized the need for a new fact, ARTHUR must now generate one such that it will provide the desired connection. This is done by choosing the default top-down index from the goal (S-HUNGER in this case) and causing it to be generated as a bottom-up index from the action. Once this is done, then specification search between the goal and action will succeed, yielding the newly augmented index, since that index will now be part of both the top-down and bottom-up indices from the goal and action. Hence adding this new index constitutes adding a new inference rule which bridges the gap between the goal and the action.

Rule 52: Choosing an improbable assumption

If an improbable assumption is needed to connect a goal and action,

then choose the default top-down index from the goal, and cause it to be generated as a bottom-up index from the action.

In the current example, the default top-down index from S-HUNGER is EAT. We now wish to cause this to be generated as part of the bottom-up index from the action of ATRANSing ants to oneself. In other words, we have a new inference that we want generated from a particular action. This is done by the same method that a story-supplied inference rules gets added to memory, as described in the previous parts of this chapter. Specifically, the memory attributes of the role fillers of the instantiated action are augmented to include the new index.

In the present example, the PLANNER attribute under JOHN and the FUNCTION attribute of ANTS are both augmented to include EAT as a bottom-up index. Hence, from the action of John ATRANSing ants to himself, ARTHUR will generate EAT as the default bottom-up index. Specification search will then yield EAT as the connective explanation index between this action and the goal of S-HUNGER.

5.6 ARTHUR output

Following is an example of ARTHUR's processing of a story which requires ARTHUR to make an improbable assumption and then use the information in that assumption as a new inference rule, in order to generate an inference which can connect the two story statements.

```
[PH: Initiation. 22-Nov-79  3:11PM]

 Yale TOPS-20 Command Processor 3A(415)-2
@RUN ARTHUR

*(ARTHUR)
Input story: *EX4

The story is:
 JOHN WAS HUNGRY. HE GOT SOME ANTS.

((S-HUNGER (ACTOR JOHN))
 (ATRANS (ACTOR JOHN) (OBJECT ANTS) (TO JOHN)))
```

ARTHUR OUTPUT	ANNOTATION

NEXT SENTENCE:	The first sentence is a
----------	statement of a goal,
JOHN WAS HUNGRY.	S-HUNGER.
*** INPUT IS GOAL:	
(S-HUNGER (ACTOR JOHN))	
NEW EXPLANATION	The goal serves as an
(S-HUNGER (ACTOR JOHN))	explanation itself. Since
ADDED TO EXPLANATION-GRAPH	there are no other goals
	in the representation,
GENERATING TOP-DOWN INDEX:	the only processing that
((EAT (ACTOR JOHN))	occurs is the generation
($RESTAURANT (ACTOR JOHN))	of the top-down predictive
($GROCER (ACTOR JOHN)))	index from the goal.
UPDATING EXPLANATION-GRAPH:	
EXPLANATION TRIPLE:	
GOAL: ((S-HUNGER	
(ACTOR JOHN))	
INVULNERABLE)	
EVENT: NIL	
INDEX:	
((EAT	
(ACTOR JOHN))	
($RESTAURANT	
(ACTOR JOHN))	
($GROCER	
(ACTOR JOHN)))	
STATUS: ACTIVE	

NEXT SENTENCE:	

HE GOT SOME ANTS.	The second sentence is an
	ATRANS action.
*** INPUT EVENT:	
(ATRANS (ACTOR JOHN)	
(OBJECT ANTS)	
(TO JOHN))	

```
GENERATING BOTTOM-UP INDEX:
((LOOK-AT (ACTOR JOHN)
          (OBJECT ANTS)))
FOR EVENT:
(ATRANS (ACTOR JOHN)
        (OBJECT ANTS)
        (TO JOHN))

PERFORMING SPECIFICATION SEARCH

*** SPECIFICATION FAILURE ***

FOUND INVULNERABLE EXPLANATION
(S-HUNGER (ACTOR JOHN))

*** FORCING INTERPRETATION ***
(S-HUNGER (ACTOR JOHN))
IS EXPLANTION FOR EVENT
(ATRANS (ACTOR JOHN)
        (OBJECT ANTS)
        (TO JOHN))

*** GENERATING
    IMPROBABLE ASSUMPTION

CHOOSING DEFAULT TOP-DOWN INDEX
FROM GOAL:
(S-HUNGER (ACTOR JOHN))

INDEX IS:
(EAT (ACTOR JOHN))

AUGMENTING BOTTOM-UP INDEX
OF EVENT:
(ATRANS (ACTOR JOHN)
        (OBJECT ANTS)
        (TO JOHN))
WITH INDEX:   (EAT)

AUGMENTING FUNCTION ATTRIBUTE
OF OBJECT:
(ANTS)
WITH INDEX:   (EAT)
```

The default function built into ARTHUR's memory says that the only use of ants is watching them, which is represented here by the LOOK-AT index.

ARTHUR then tries to explain this action in terms of the previous S-HUNGER goal. This fails, but ARTHUR cannot supplant that goal because it is invulnerable, that is, it was explicitly stated in the story, and was not an inference.

Hence, ARTHUR now forces an interpretation: it assumes that the stated S-HUNGER goal must be the explanation for the ATRANS ANTS action via some previously unknown inference path.

ARTHUR assumes that the inference path between the goal and the action is indexed by the EAT plan.

Accordingly, ARTHUR adds this plan as a bottom-up index from the ATRANS ANTS action. It does this by augmenting the memory attributes of the elements of this action.

In this case, the OBJECT slotfiller, ANTS, has its FUNCTION attribute augmented to include the EAT index.

```
*** GENERATING
NEW BOTTOM-UP INDEX:
(EAT (ACTOR JOHN)
     (OBJECT ANTS))
FROM EVENT:
(ATRANS (ACTOR JOHN)
        (OBJECT ANTS)
        (TO JOHN))

*** RE-EXPLAINING INPUT

PERFORMING SPECIFICATION SEARCH

FOUND EXPLANATION INDEX:
(EAT (ACTOR JOHN)
     (OBJECT ANTS))

UPDATING EXPLANATION-GRAPH:
EXPLANATION TRIPLE:
  GOAL: ((S-HUNGER
           (ACTOR JOHN))
         INVULNERABLE)
     EVENT:
        (ATRANS (ACTOR JOHN)
                (OBJECT ANTS)
                (TO JOHN))
     INDEX:
        ((EAT
          (ACTOR JOHN)
          (OBJECT ANTS)))
     STATUS: ACTIVE

*** FINAL EXPLANATION-GRAPH

EXPLANATION TRIPLE:
  GOAL: ((S-HUNGER
           (ACTOR JOHN))
         INVULNERABLE)
     EVENT:
        (ATRANS (ACTOR JOHN)
                (OBJECT ANTS)
                (TO JOHN))
     INDEX:
        ((EAT
          (ACTOR JOHN)
          (OBJECT ANTS)))
     STATUS: ACTIVE

*** PROCESSING COMPLETED
*** READY FOR QUESTIONS
```

Now ARTHUR generates the newly-augmented EAT index from the ATRANS ANTS action.

This will result in the action being explained by the S-HUNGER goal, since now specification search will be able to succeed: the EAT index will be present in both the top-down and the bottom-up indices. Hence the new interpretation of the action is "forced" by the addition of this bottom-up index.

The final explanation-graph represents the ATRANS action as being performed in service of satisfying hunger, by the plan of eating the ants.

```
Input question:*EX4Q1

QUESTION:
 WHY DID JOHN GET SOME ANTS?

ANSWER:
 MAYBE BECAUSE HE WANTED TO EAT THEM.

Input question:*EX4Q2


QUESTION:
 WHY WAS JOHN HUNGRY?

ANSWER:
 THE STORY SAID THAT HE WAS HUNGRY.

Input question:*#

[PH: Termination. 22-Nov-79  3:12PM. PS:<GRANGER.TH>ANT2.LOG.1]
```

5.7 Summary

New facts about the world can be specified in a story, and can be represented as inference rules in an understander's memory. Understanding a story which contains new facts requires the reader to recognize these facts, incorporate them into memory and use them to generate new inferences to understand the rest of the story. These processes were specified and the rules and knowledge needed for their implementation was presented. These rules were used by ARTHUR to enable it to understand stories containing new inference rules.

CHAPTER 6

HOW ARTHUR WORKS

6.1 Introduction

The previous chapters have shown some of the algorithms by which ARTHUR maintains and updates a consistent and parsimonious representation of stories. This chapter will examine in detail the way these algorithms work together in ARTHUR to form a coherent error-correcting understanding system.

There are a number of different aspects of ARTHUR that are addressed here. ARTHUR's overall understanding process always attempts to incorporate new story statements into the story representation. The understanding process consists of three basic "steps", or subprocesses, which are always performed in the same order. These subprocesses are: (1) performing specification search on indexed inferences, (2) supplanting previous inferences, and (3) forcing an interpretation. Each subprocess is applied only when the one before it has failed to incorporate a new input.

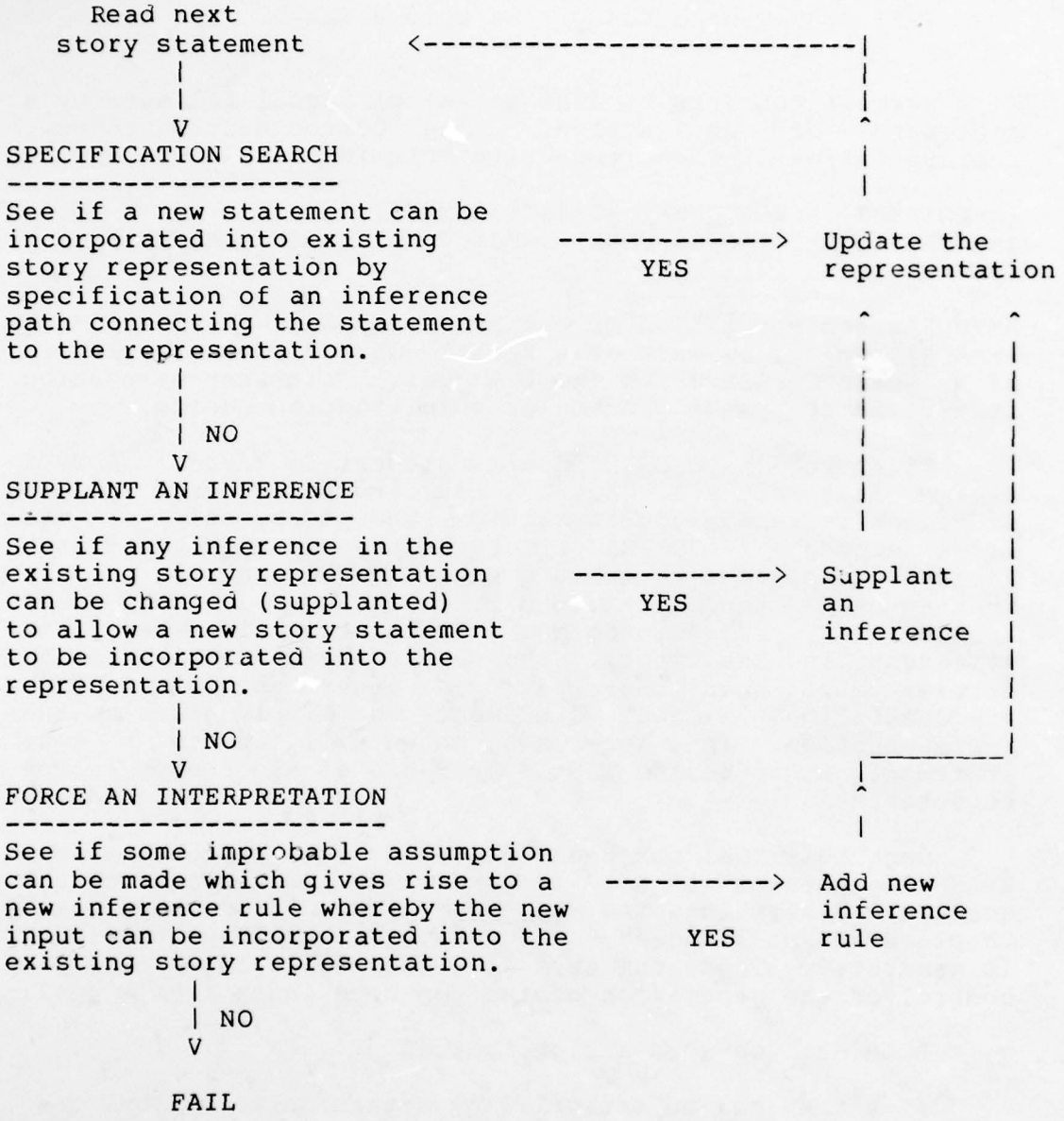
ARTHUR recognizes three different classes of input: actions and states; goals; and new inference rules. The application of the above three subprocesses differs depending on which type of input ARTHUR reads. Once ARTHUR has successfully explained all the inputs or statements in a story, it can answer questions about the story by referencing the representation it has built of the story. ARTHUR's question-answering and generation processes are also described in this chapter.

6.2 Overview of the understanding process

ARTHUR's understanding algorithm essentially consists of reading story statements, one at a time, and incorporating each one into its representation of the story. "Incorporating" an input into the representation means explaining the story events in terms of the stated or inferred goals in the story. The notion of explaining story events is described in Chapter 1. That chapter also gives a flowchart, containing a very sketchy outline of the flow of the three basic processes ivolved in incorporating a new input into a story representation. Those three processes are invoked in the same order for each new story input: (1) specification search, (2) supplanting an inference, (3) forcing an interpretation.

Each of these processes is used only when the previous one has failed. Hence, ARTHUR only supplants an inference if a new input cannot be incorporated by specification search. Similarly, ARTHUR only forces an interpretation if supplanting fails to incorporate the input. For any particular story statement, only one of the three processes will be successful, although each may be attempted, in the process of incorporating that statement into the representation. The following flow diagram outlines the processing that ARTHUR uses to incorporate a new input. The following sections will give detailed descriptions of each of ARTHUR's three major understanding subprocesses.

ARTHUR's processing algorithm



[Figure 20]

6.3 Specification search

Consider the following:

[75] John wanted money. He sold a watch.

This example consists of a statement of a goal followed by a statement of an action. The Conceptual Dependency representation of these two statements is as follows:

(A-POSSESS (ACTOR JOHN) (OBJECT MONEY))
(MUTUAL-ATRANS (ACTOR JOHN) (OBJECT WATCH) (FOR MONEY))

ARTHUR's representation of the story contains the "sell" (or ATRANS) action, as part of a BARGAIN-OBJECT plan, in service of an ACHIEVE-POSSESSION (MONEY) goal. This representation is arrived at by the process of specification search.

First the "A-POSSESS" goal statement is read. ARTHUR has a list of all known goals in its memory. A goal statement is recognized as such by the membership of its first element (A-POSSESS in this case) in this goal list. Having recognized a goal statement, ARTHUR checks to see whether any other statements or inferences have been added to the story representation. Since this is the first statement in the story, the story representation has no entries yet. When there are no other entries in the representation, a goal statement is simply added to the representation. In a later section we will see how a goal statement is processed in the presence of a non-empty story representation.

Once this goal has been added to the representation, ARTHUR generates a set of predictive inferences from the goal, by generating the top-down index from the goal. Chapters 2 and 3 showed the rules by which a top-down index is generated. Those chapters gave the following rules in control of the generation of the top-down index from a goal:

Rule 53: Top-down script indices

If a goal can be satisfied by a particular script,
Then that script is among the top-down indices
from the goal.

Rule 54. Top-down plan indices

If a goal can be satisfied by a particular plan,
Then that plan is among the top-down indices from the goal.

ARTHUR's implementation of these rules consists of encoding all such top-down indices under the "TDI" (Top-Down Index) property of the goal in its memory. Under the TDI property of A-POSSESS, ARTHUR has the following list:

ASK
INVOKE-THEME
INFORM-REASON
BARGAIN-OBJECT
BARGAIN-FAVOR
THREATEN
OVERPOWER
STEAL

This list is simply the list of PERSUADE package plans described by Schank and Abelson [1977]. ARTHUR considers all of these to be valid predictions as to the plan an actor might use to satisfy an A-POSSESS goal. Each of these predictive indices is instantiated when it is generated as part of the top-down index from John's goal. That is, since ARTHUR knows that this is specifically one of John's goals, and furthermore since John specifically wants to possess money, each of the plans is instantiated to specify that John is the PLANNER and the OBJECT of the plan is MONEY. Instantiation of indices is performed by applying the INSTANTIATE function to each element of the index.

The INSTANTIATE function adds slots and slotfillers to the index, corresponding to the slots and slotfillers in the goal. Hence, for each of the plans in the above list, INSTANTIATE will add the slots PLANNER and OBJECT, and and slotfillers JOHN and MONEY, respectively. The resulting instantiated top-down index is as follows:

```
(ASK (ACTOR JOHN) (OBJECT MONEY))
(INVOKE-THEME (ACTOR JOHN) (OBJECT MONEY))
(INFORM-REASON (ACTOR JOHN) (OBJECT MONEY))
(BARGAIN-OBJECT (ACTOR JOHN) (OBJECT MONEY))
(BARGAIN-FAVOR (ACTOR JOHN) (OBJECT MONEY))
(THREATEN (ACTOR JOHN) (OBJECT MONEY))
(OVERPOWER (ACTOR JOHN) (OBJECT MONEY))
(STEAL (ACTOR JOHN) (OBJECT MONEY))
```

Having generated an instantiated top-down index from the goal, ARTHUR now adds the goal and the index to the story representaion.

ARTHUR's story representation is an explanation graph. We have so far seen pictorial versions of explanation graphs in this thesis. ARTHUR's internal representation of an explanation graph consists of a set of "triples", each containing a goal, and an action that was performed in service of that goal via some plan or script index. The syntax of explanation graphs in ARTHUR can be expressed as follows:

```
EXPLANATION-GRAPH = <TRIPLE1 + TRIPLE2 ...>

TRIPLE            = <<GOAL>
                     <ACTION1>
                     <INDEX-LIST1>
                     <STATUS>
                     <ACTION2>
                     <INDEX-LIST2>
                     <STATUS>
                      ...>

GOAL              = <Goals as in Schank and Abelson [1977]>

ACTION            = <Conceptual Dependency actions>

INDEX-LIST        = <INDEX1 + INDEX2 ...>

INDEX             = <Plan, script and causal state-change indices,
                     as described in Chapters 2 and 3 of this thesis>

STATUS            = <ACTIVE | SUPPLANTED>
```

In the current example, ARTHUR has read a single goal statement, and generated a top-down index from it. There is as yet no event explained by the goal as part of any of the predicted plans. The story representation so far is as follows:

```
TRIPLE:
  GOAL:   (A-POSSESS (ACTOR JOHN) (OBJECT MONEY))
  EVENT:  NIL
  INDEX:  ((ASK (ACTOR JOHN) (OBJECT MONEY))
           (INVOKE-THEME (ACTOR JOHN) (OBJECT MONEY))
           (INFORM-REASON (ACTOR JOHN) (OBJECT MONEY))
           (BARGAIN-OBJECT (ACTOR JOHN) (OBJECT MONEY))
           (BARGAIN-FAVOR (ACTOR JOHN) (OBJECT MONEY))
           (THREATEN (ACTOR JOHN) (OBJECT MONEY))
           (OVERPOWER (ACTOR JOHN) (OBJECT MONEY))
           (STEAL (ACTOR JOHN) (OBJECT MONEY)))
  STATUS: ACTIVE
```

Now the second statement is read:

(MUTUAL-ATRANS (ACTOR JOHN) (OBJECT WATCH) (FOR MONEY))

The input to ARTHUR is in this Conceptual Dependency form, not in English. English sentences are generated from this Conceptual Dependency action at the time the action is read. ARTHUR's generation algorithm is discussed in a later section of this chapter.

The first step in explaining an event is to generate a set of bottom-up inferences from the event. For every CD action, ARTHUR has a set of rules which look up particular attributes of particular slotfillers in an instantiated action. For instance, MUTUAL-ATRANS itself gives rise to a BARGAIN-OBJECT inference, and a D-CONT precondition inference. Both of these indices are listed under the "BUI" (Bottom-Up Index) attribute of the entry for MUTUAL-ATRANS in ARTHUR's memory.

In addition, the rule under MUTUAL-ATRANS looks up the FUNCTION of the FOR slot and the PLANNER of the actor slot of the instantiated action, which are MONEY and JOHN, respectively. In this case, there are no entries under the PLANNER attribute of the ACTOR slotfiller JOHN, nor the FUNCTION attribute of the FOR slotfiller MONEY. Hence the complete bottom-up index from the instantiated MUTUAL-ATRANS is the plan BARGAIN-OBJECT, and the precondition D-CONT. Precondition indices are generated as a separate index from the "pure" bottom-up index plans, in a variable called PC-BUI. We will see the operation of Precondition Bottom-Up Indices in the next section.

As with top-down indices, ARTHUR instantiates bottom-up indices from actions corresponding to the slots and slotfillers in the instantiated action. The bottom-up index from this MUTUAL-ATRANS action is as follows:

(BARGAIN-OBJECT (ACTOR JOHN) (OBJECT MONEY) (BARGAINED WATCH))

This means that John planned to trade his watch for money.

Now ARTHUR attempts to incorporate this event into the story representation. This is performed by the process of specification search, which consists of intersecting the top-down index from a goal in the representation with the bottom-up index from the action to be explained. This process is performed by first intersecting just the uninstantiated versions of the indices, that is, the first element in each index. Then each of the roles and rolefillers of the intersection set are matched with each other. Whenever one index contains a role not contained by the other index, that role is added to the index that is missing it. For example, the fact that John used a watch in his bargaining was not predicted from his goal. Nonetheless, it is specified by his action, and hence is included in the inferred bottom-up plan. Hence this information further specifies the inference about John's plan for satisfying his goal.

The result of specification search is called the explanation index. This index represents the complete inference path from the action to the goal. In this case, specification search yields the index

(BARGAIN-OBJECT (ACTOR JOHN) (OBJECT MONEY) (BARGAINED WATCH))

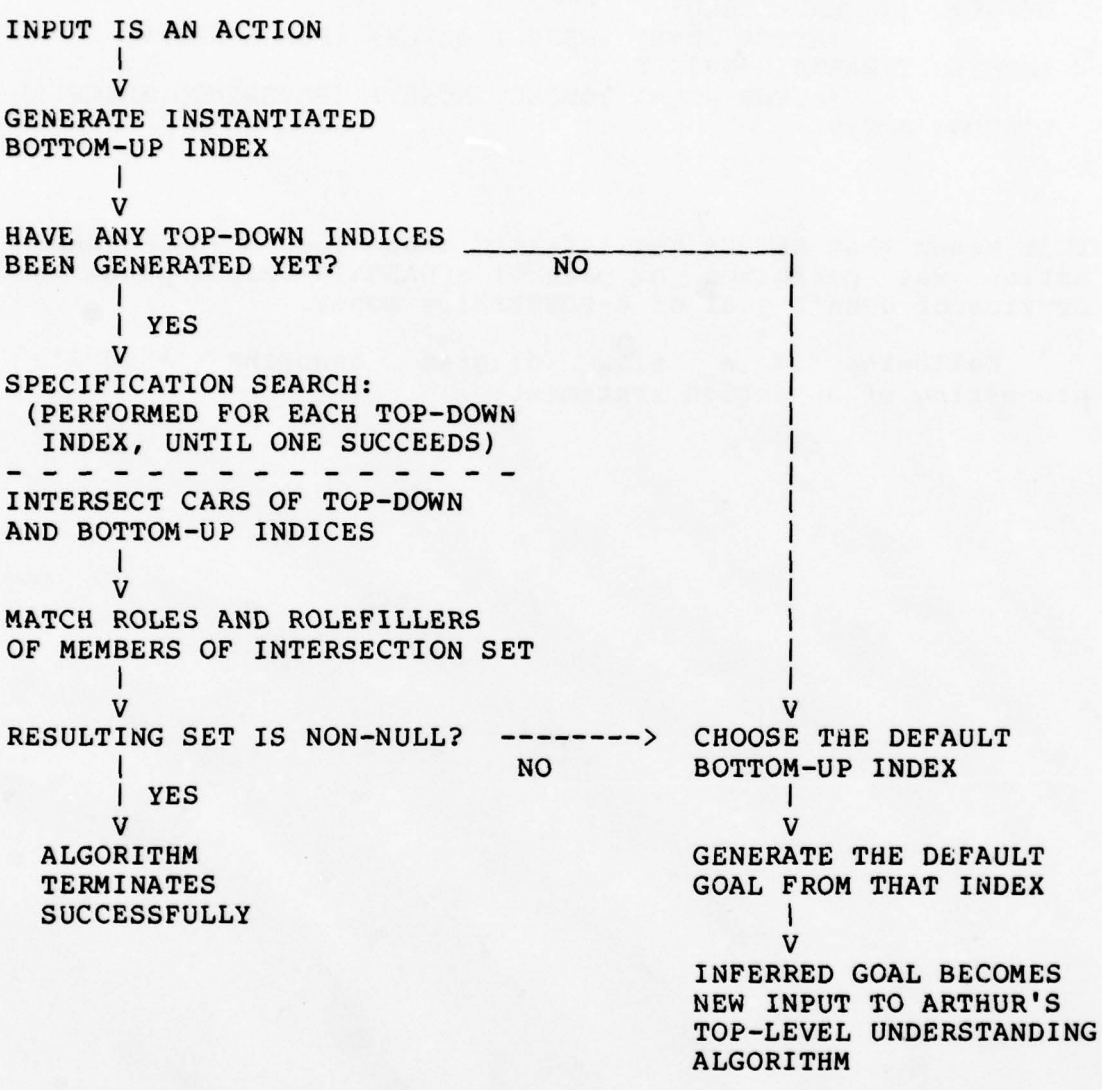
The story representation is now updated to represent this index as the connection between the action and the goal, as follows:

```
TRIPLE:
  GOAL:    (A-POSSESS
               (ACTOR JOHN) (OBJECT MONEY))
  EVENT:   (MUTUAL-ATRANS
               (ACTOR JOHN) (OBJECT WATCH) (FOR MONEY))
  INDEX:   ((BARGAIN-OBJECT
               (ACTOR JOHN) (OBJECT MONEY) (BARGAINED WATCH)))
  STATUS: ACTIVE
```

This means that ARTHUR has inferred that the MUTUAL-ATRANS action was performed as part of a BARGAIN-OBJECT plan, in service of John's goal of A-POSSESSing money.

Following is a flow diagram denoting ARTHUR's processing of an action statement:

ARTHUR's event incorporation algorithm



[Figure 21]

6.4 Supplanting inferences

The above flow diagram outlines ARTHUR's processing when its input is a statement of an action. By tracing the diagram's paths, it is seen that the algorithm terminates successfully if and only if the following two criteria are met:

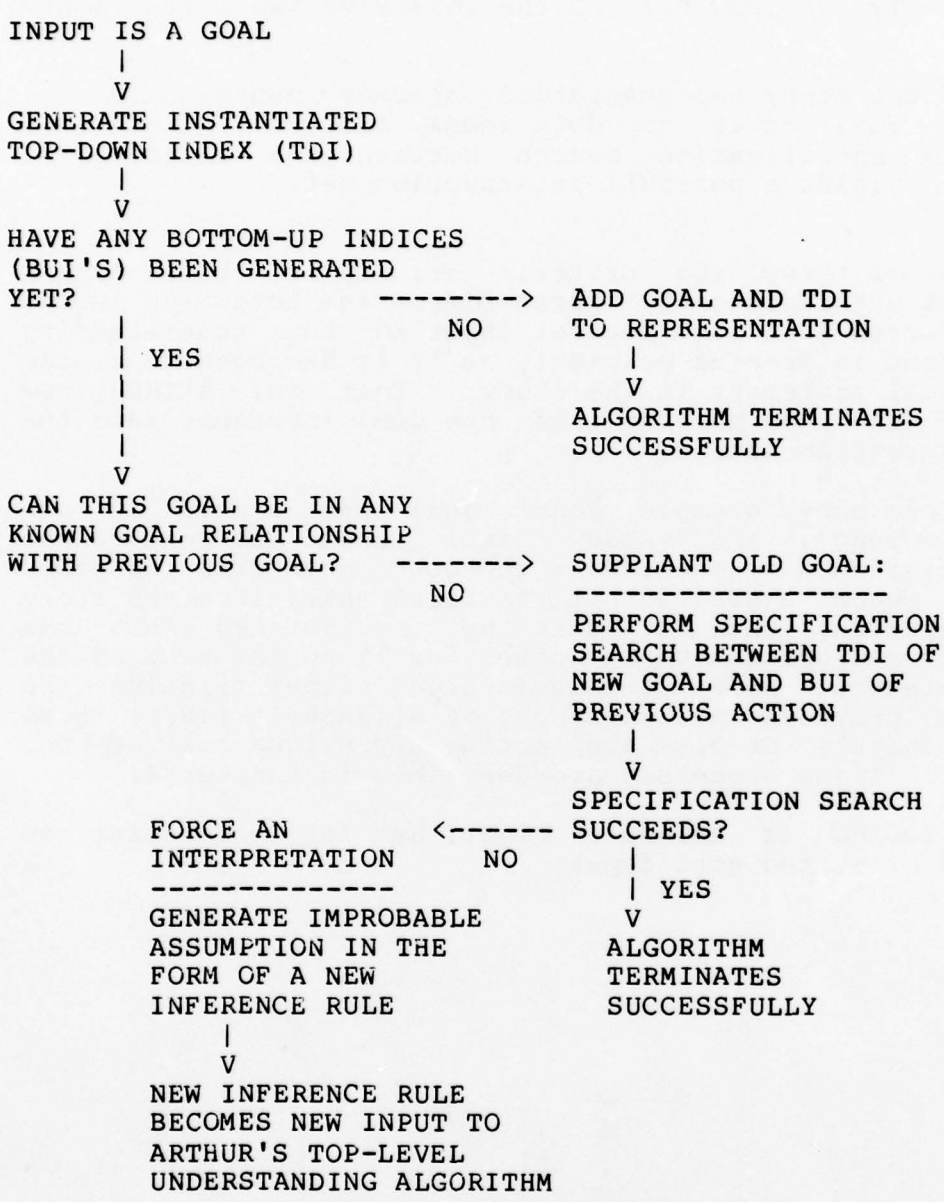
1 - the story representation already contains a goal and its top-down index, and
2 - specification search between the indices yields a non-null intersection set.

If either of these two criteria are false, then ARTHUR generates a default goal inference from the bottom-up index. That inferred goal then becomes input to the understanding system, and is treated precisely as if it had been generated from a goal statement in the story. That is, ARTHUR now attempts to incorporate this new goal inference into the existing representation.

In the above example, John's goal was stated in the first sentence, and ARTHUR simply added the goal to the representation and generated a top-down index from the goal. However, when a goal is not the first entry into the story representation, then it must be incorporated into the existing representation by connecting it to the rest of the representation. Briefly, this involves either relating the goal to previous goals via one of Wilensky's [1978] "goal relationships"; or else supplanting a previous goal by the new goal. These processes are described in Chapter 4.

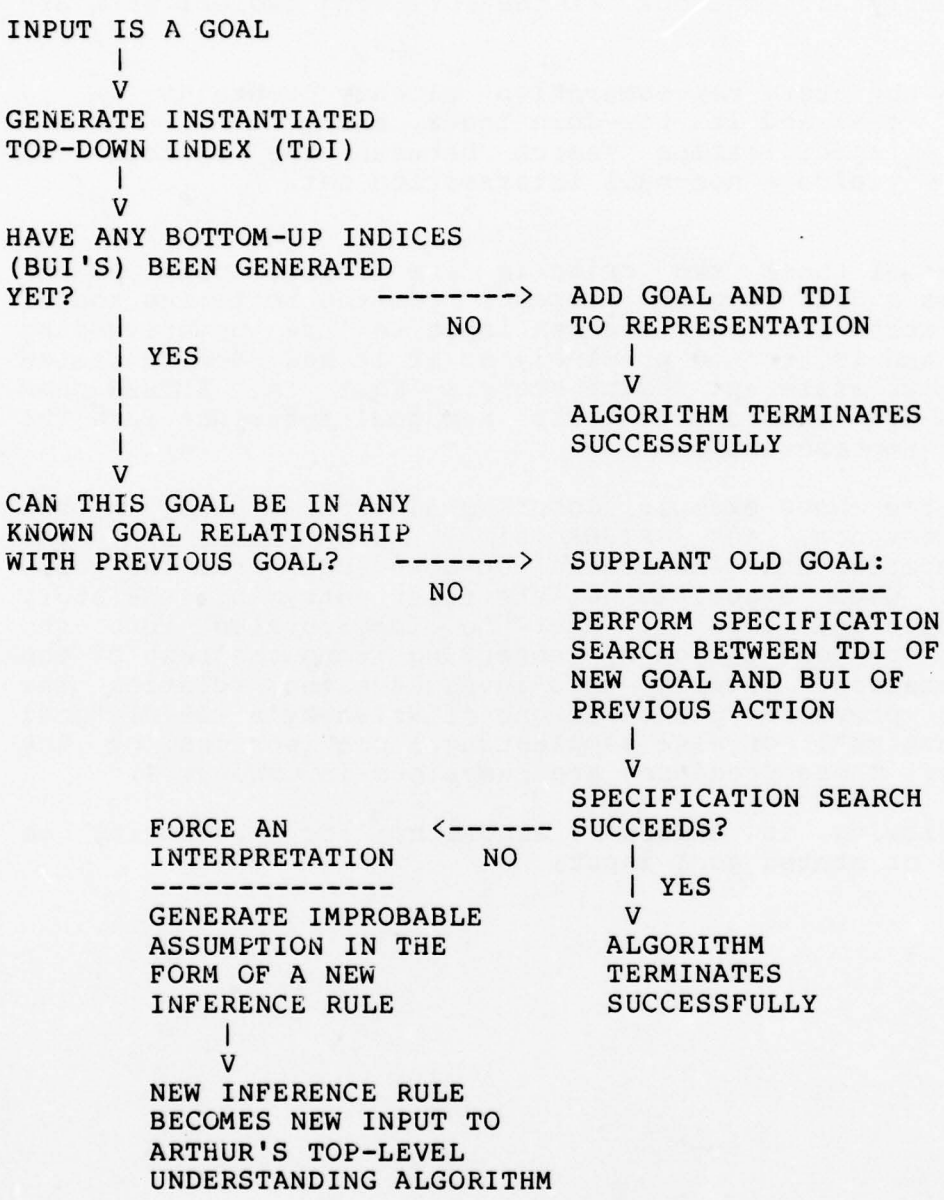
Following is ARTHUR's algorithm for processing an inferred or stated goal input:

ARTHUR's goal incorporation algorithm



[Figure 22]

ARTHUR's goal incorporation algorithm



[Figure 22]

The subprocesses in the above algorithm are those of generating bottom-up and top-down indices, and performing specification search. These processes are implemented as previously described in the event-incorporation algorithm The only new process introduced in the above algorithm is that of determining when two goals are in a goal relationship with each other. The rules underlying this process are described in Wilensky [1978]. The actual implementation in ARTHUR is done by adding a set of rules corresponding to Wilensky's, each of which check properties of the goals involved. Currently only four of Wilensky's rules are implemented, and ARTHUR will therefore sometimes miss an implied goal relationship. However, this is only a side issue in this thesis, so this is not considered to be a crucial omission from ARTHUR's implementation.

Recall the following example from Chapter 4:

[76] John went to the gas station. He robbed the station.

The first statement is an action:

(PTRANS (ACTOR JOHN) (OBJECT JOHN) (TO GAS-STATION))

The bottom-up index from this action arises from the LOCATION-INDICES attribute of the memory entry for GAS-STATION, in addition to the precondition index listed in ARTHUR's entry under the BUI attribute of PTRANS. The index is the $GAS-STATION script, and the Precondition index is D-PROX.

According to the event incorporation algorithm given previously, this bottom-up index is generated, and then any existing top-down index will be intersected with it via specification search. In this case, the action is the first story statement, so there is no existing top-down index in the representation yet. Hence the algorithm specifies the generation of a default goal. Every index has an associated default goal inference. That of $GAS-STATION is A-POSSESS (GAS). The instantiated index and goal are added to the representation. The representation now is as follows:

```
TRIPLE:
  GOAL:    (A-POSSESS
               (ACTOR JOHN) (OBJECT GAS))
  EVENT:   (PTRANS
               (ACTOR JOHN) (OBJECT JOHN) (TO GAS-STATION))
  INDEX:   (($GAS-STATION
               (ACTOR JOHN)))
  STATUS: ACTIVE
```

The second story statement is another action, with a constrained bottom-up index. As described in Chapter 3, certain English words imply not just a Conceptual Dependency action, but also the plan that the action is part of. For example, "demand" is an MTRANS as part of a THREATEN plan. Similarly, "rob" is an ATRANS as part of either a THREATEN, OVERPOWER or STEAL plan. The representation of these words in a sentence reflects this specified plan, as follows:

```
(ATRANS (ACTOR JOHN) (OBJECT MONEY) (FROM GAS-STATION)
        (BUI (THREATEN OVERPOWER STEAL)))
```

ARTHUR generates the bottom-up index from this action by instantiating the indices in the "BUI" role specified in the action. Then ARTHUR performs specification search between the top-down index from the previously inferred A-POSSESS (GAS) goal, and the bottom-up index from this action. The top-down index from A-POSSESS (GAS) currently consists of the single index $GAS-STATION, since that is what ARTHUR inferred as John's plan to satisfy that goal. (This is a constrained top-down index, as described in Chapters 2 and 4.)

The specification search fails to yield an explanation index in this example: there is no intersection set between these top-down and bottom-up indices. According to the event incorporation algorithm, ARTHUR generates a new default goal inference from the bottom-up index of the ATRANS action. The default index is always the first one in the list, which in this case is THREATEN, and the default goal is A-POSSESS (MONEY). This new goal is then processed according to the goal incorporation algorithm. The explanation-graph representation at this point is as follows:

```
TRIPLE:
  GOAL:    (A-POSSESS
               (ACTOR JOHN) (OBJECT GAS))
  EVENT:   (PTRANS
               (ACTOR JOHN) (OBJECT JOHN) (TO GAS-STATION))
  INDEX:   (($GAS-STATION
               (ACTOR JOHN)))
  STATUS: ACTIVE

TRIPLE:
  GOAL:    (A-POSSESS
               (ACTOR JOHN) (OBJECT MONEY))
  EVENT:   (ATRANS
               (ACTOR JOHN) (OBJECT MONEY) (FROM GAS-STATION)
               (BUI (THREATEN OVERPOWER STEAL)))
  INDEX:   ((THREATEN
               (ACTOR JOHN) (VICTIM GAS-STATION) (OBJECT MONEY)))
  STATUS: ACTIVE
```

This is a disconnected explanation, as described in Chapter 4. ARTHUR attempts to connect the explanation, according to the goal incorporation algorithm above.

First, the goal is checked against the previous A-POSSESS (GAS) goal to see if any goal relationship could exist. None is found by any of Wilensky's [1978] rules. Next ARTHUR supplants the initial goal by the new one. This involves performing specification search between the top-down index from the new A-POSSESS (MONEY) goal and the unconstrained bottom-up index from the PTRANS action. The bottom-up index from the PTRANS was "constrained" to include only the $GAS-STATION index, at the time that ARTHUR inferred that that index was John's plan for the PTRANS action. Unconstraining the index is implemented by re-generating the bottom-up index from the action. Recall that the unconstrained index from the PTRANS action consisted of the $GAS-STATION script index and the D-PROX precondition index.

Recall that we said that precondition indices were generated separately from "pure" plan and script indices, in the PC-BUI variable. This is because ARTHUR first checks all "pure" indices via specification search before checking precondition indices. This is because precondition indices will match top-down indices much more flexibly than other indices. A precondition index will match any top-down plan index which either has that precondition index as one of its STEP attributes, or any top-down plan index which can include the roles of the precondition as one of the roles in the plan. This "flexible" matching is described in Chapters 2 and 4.

In the current example, the $GAS-STATION bottom-up script index fails to match the THREATEN top-down index. The specification search algorithm then checks the precondition indices to see if they might match. The precondition index matching process within the specification search algorithm is as follows:

Precondition matching algorithm

IF a BU precondition is a member of the set of plans contained in the STEPS attribute of a particular top-down index,
OR every rolefiller in the BU precondition appears in the top-down index,
OR every rolefiller in the BU precondition satisfies the feature constraints of a null role in the top-down index,

THEN the BU precondition matches the top-down index.

In the current example, the bottom-up precondition index and the top-down index are as follows:

(DPROX (ACTOR JOHN) (LOC GAS-STATION))

(THREATEN (ACTOR JOHN) (VICTIM GAS-STATION) (OBJECT MONEY))

By the precondition matching algorithm given above, the DPROX will match the THREATEN, since both of the rolefillers in the DPROX index appear in the THREATEN index. Hence the DPROX can be inferred to be a precondition for John's THREATEN plan.

Specification search succeeds, resulting in the PTRANS action being re-explained in terms of the new A-POSSESS (MONEY) goal. ARTHUR then marks the initial A-POSSESS (GAS) goal as having been supplanted, and adds the PTRANS action to the new goal's triple, resulting in the following explanation graph:

```
TRIPLE:
  GOAL:    (A-POSSESS
               (ACTOR JOHN) (OBJECT GAS))
  EVENT:   (PTRANS
               (ACTOR JOHN) (OBJECT JOHN) (TO GAS-STATION))
  INDEX:   (($GAS-STATION
               (ACTOR JOHN)))
  STATUS: SUPPLANTED

TRIPLE:
  GOAL:    (A-POSSESS
               (ACTOR JOHN) (OBJECT MONEY))
  EVENT:   (ATRANS
               (ACTOR JOHN) (OBJECT MONEY) (FROM GAS-STATION)
               (BUI (THREATEN OVERPOWER STEAL)))
  INDEX:   ((THREATEN
               (ACTOR JOHN) (VICTIM GAS-STATION) (OBJECT MONEY)))
  STATUS: ACTIVE
  EVENT:   (PTRANS
               (ACTOR JOHN) (OBJECT JOHN) (TO GAS-STATION))
  INDEX:   ((DPROX
               (ACTOR JOHN) (LOC GAS-STATION)))
  STATUS: ACTIVE
```

This corresponds to the pictorial explanation graph given in Figure 56 in Chapter 4.

6.5 Forcing an interpretation

According to the goal incorporation algorithm depicted in Figure 22, when supplanting an inference fails to yield a connected explanation, ARTHUR generates an improbable assumption which contains a set of new inference rules which can connect a previously unconnected goal-action pair. Chapter 5 presented rules for accomplishing this. Those rules outline a method of choosing the default top-down inference from the goal and adding this inference to the rolefillers of the action, via the process of bottom-up index augmentation. Once a new inference rule has been generated in this way, and added to memory by index augmentation, then ARTHUR's normal inference generation mechanisms will result in the use of the information in the new rule in generating any future inferences which refer to the augmented memory entries.

As specified in Chapter 5, index augmentation is performed by inserting the new index at the front of the appropriate property under the specified memory element. For example, consider the following statement, which contains a story-supplied inference rule:

[77] If you push the button on this death ray, and then someone walks close to it, then they will die.

This story specifies a three-step plan for achieving a goal of someone's HEALTH state becoming (-10). The three steps in the plan are:

1 - the planner pushes the button on the death ray
2 - the victim walks to the proximity of the ray
3 - the victim's health state changes negatively

These three steps are represented by the following three indices:

```
(PHYS-CONTACT (OBJECT BUTTON))
(DPROX (OBJECT VICTIM) (TO DEATH-RAY))
(CHANGE-HEALTH (VAL NEG) (OBJECT VICTIM))
```

ARTHUR's processing of this statement will result in the STEPS attribute of each of these three indices being augmented by the addition of each of the other two steps. For example, the STEPS attribute of the CHANGE-HEALTH index will now be as follows:

```
CHANGE-HEALTH
-------------
   STEPS:
   ------
   [(STEP
     (PHYS-CONTACT
       (ROLES
        (PLANNER . PLANNER)
        (OBJECT (QUOTE BUTTON)))))
    (STEP
     (DPROX
       (ROLES
        (PLANNER . PLANNER)
        (OBJECT . OBJECT)
        (LOC (QUOTE WEAPON)))))]
```

This represents the fact that the two indices PHYS-CONTACT and DPROX will henceforth be recognized as being potential preconditions for the CHANGE-HEALTH index. Similarly, the other two indices each have their respective counterparts added to their STEPS attributes:

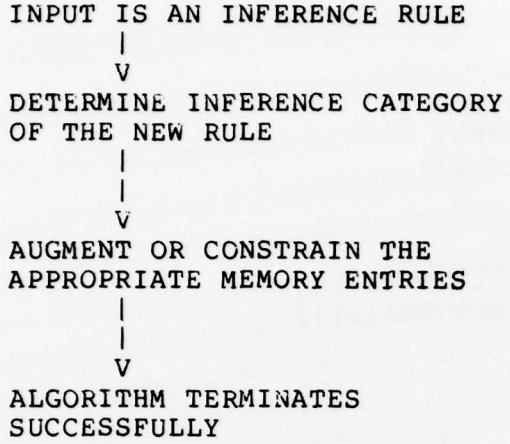
```
DPROX
-----
   STEPS:
   ------
   [(PLAN
     (CHANGE-HEALTH
       (ROLES
        (VAL (QUOTE -10))
        (PLANNER . PLANNER)
        (OBJECT . OBJECT))))
    (STEP
     (PHYS-CONTACT
       (ROLES
        (PLANNER . PLANNER)
        (OBJECT (QUOTE BUTTON)))))]

PHYS-CONTACT
------------
   STEPS:
   ------
   [(PLAN
     (CHANGE-HEALTH
       (ROLES
        (VAL (QUOTE -10))
        (PLANNER . PLANNER)
        (OBJECT . ?IR1))))
    (STEP
     (DPROX
       (ROLES
        (PLANNER . PLANNER)
        (OBJECT . ?IR1)
        (LOC (QUOTE WEAPON)))))]
```

Note that the attribute list specifies either PLAN or STEP for every entry in the list. This represents the difference between the final step in a plan, and the preconditions that must have occurred before that final step. In this example, the CHANGE-HEALTH is specified as a plan which has two preconditions: PHYS-CONTACT and DPROX. From now on, whenever ARTHUR generates either of these latter two indices, it will infer that that index may be a precondition for a CHANGE-HEALTH plan. Hence, in addition, the default goal inference from either of these two indices is the default goal inference from the CHANGE-HEALTH plan, namely that of a HEALTH state of (-10).

Following is a flowchart of ARTHUR's algorithm for incorporating a new inference rule into its memory:

ARTHUR's inference rule incorporation algorithm



[Figure 23]

6.6 Input and output

ARTHUR's understanding ability depends in part on its ability to read input stories and input questions, and to arrive at and generate output answers to the questions. Most of the algorithms given in this thesis for ARTHUR's understanding processes take input and output in the form of Conceptual Dependency representations. This section will briefly examine the three processes of question answering, parsing, and generation, as they are implemented in ARTHUR.

6.6.1 Question answering

ARTHUR can answer questions about stories it has read and understood. The question-answering process implemented in ARTHUR is not intended as a full-scale theory of question-answering. The program borrows from rules suggested by Lehnert [1975], in order to provide ARTHUR with a minimal ability to demonstrate its understanding. ARTHUR only recognizes questions about the goals and plans underlying characters' actions.

ARTHUR goes into a special "question answering mode" after it has processed a story. (A switch can be set to make ARTHUR go into question mode after every sentence of a story). The format of questions is identical to the format of regular story statements. When in question mode, ARTHUR interprets an event statement as a question about the goals underlying that event. For example, consider the following actual input and output from ARTHUR:

```
INPUT STORY:
 JOHN WENT TO THE GAS STATION. HE ROBBED THE STATION.

INPUT QUESTION:
 WHY DID JOHN GO TO THE GAS STATION?

OUTPUT ANSWER
 AT FIRST I THOUGHT IT WAS BECAUSE HE WANTED TO GET SOME
GASOLINE, BUT ACTUALLY IT'S BECAUSE HE WANTED TO GO THERE
TO STEAL MONEY.
```

The English generation algorithm will be discussed in the next section. For now consider the following CD versions of the above input and output:

```
INPUT STORY:
((PTRANS (ACTOR JOHN) (OBJECT JOHN) (TO GAS-STATION))
 (ATRANS (ACTOR JOHN) (OBJECT MONEY) (TO JOHN) (FROM GAS-STATION)
         (BUI (THREATEN OVERPOWER STEAL))))

INPUT QUESTION:
(PTRANS (ACTOR JOHN) (OBJECT JOHN) (TO GAS-STATION))

OUTPUT ANSWER:
(((A-POSSESS (ACTOR JOHN) (OBJECT GASOLINE))
  ($GAS-STATION (ACTOR JOHN)))
 ((A-POSSESS (ACTOR JOHN) (OBJECT MONEY))
  (DELTA-PROX (ACTOR JOHN) (TO GAS-STATION))))
```

The "output answer" consists of a list of all the goals and indices that served as the PTRANS action's explanation, during the course of understanding the story. This list is an ordered "trace" or "history" of the explanations for this action, beginning with any supplanted explanations and ending up with the final explanation. This can be seen from the final explanation graph for the story, as shown previously:

```
TRIPLE:
  GOAL:    (A-POSSESS
               (ACTOR JOHN) (OBJECT GAS))
* EVENT:   (PTRANS
               (ACTOR JOHN) (OBJECT JOHN) (TO GAS-STATION))
  INDEX:   (($GAS-STATION
               (ACTOR JOHN)))
  STATUS: SUPPLANTED

TRIPLE:
  GOAL:    (A-POSSESS
               (ACTOR JOHN) (OBJECT MONEY))
  EVENT:   (ATRANS
               (ACTOR JOHN) (OBJECT MONEY) (FROM GAS-STATION)
               (BUI (THREATEN OVERPOWER STEAL)))
  INDEX:   ((THREATEN
               (ACTOR JOHN) (VICTIM GAS-STATION) (OBJECT MONEY)))
  STATUS: ACTIVE

* EVENT:   (PTRANS
               (ACTOR JOHN) (OBJECT JOHN) (TO GAS-STATION))
  INDEX:   ((DPROX
               (ACTOR JOHN) (LOC GAS-STATION)))
  STATUS: ACTIVE
```

From this graph, it can be seen that the PTRANS action appears twice, initally in a supplanted triple and finally

in an active triple. The above answer means that ARTHUR first explained the PTRANS action as being part of a GAS-STATION script in service of a goal of getting gas, but then supplanted that inference in favor of the inference that the action was actually part of a DPROX precondition to a THREATEN plan in service of a goal of getting money.

The question answering algorithm essentially consists of searching through the final explanation graph for each goal triple in which the event appears. The algorithm keeps a list of the goal and index in each such triple, and the resulting list comprises the answer to the question.

6.6.2 Parsing

ARTHUR's understanding algorithms are largely language independent, in that they will in theory work on any canonical meaning representation of any natural language inputs. All input and output from ARTHUR is in the form of Conceptual Dependency representations of actions and goals. Many of the stories that ARTHUR understands were given as English input to a parser called CA (Conceptual Analyzer) written by Larry Birnbaum and described in Birnbaum and Selfridge [1979].

CA was able to output Conceptual Dependency representations for most of the events and goals that ARTHUR recognizes. Inference rule input such as "John only eats macrobiotic" were not successfully parsed by CA. Statements containing constrained bottom-up indices such as "John robbed the gas station" can be handled by CA, but the output is not identical to that recognized by ARTHUR. We assert that the output is isomorphic, although not identical: I was able to write a "transformation" program which took all of CA's output and put it into ARTHUR-readable form.

6.6.3 Generation

ARTHUR's generation algortithm is phrase-oriented, and is dependent on features of whatever larger context contains the representation being generated. ARTHUR's generator is essentially a "micro" version of the one which is responsible for the English output in all of the examples in Wilensky [1978]. As mentioned by Wilensky [1978], I designed and wrote the first version of that generator, and Rod McGuire is responsible for enormously extending and generalizing my initial attempt.

Every Conceptual Dependency memory element that can be generated contains a set of rules which test its own surrounding context and certain features of whatever has

been previously generated. Based on this information, an English phrase is generated. Currently, such phrases can be anywhere from zero to six words in length. (A phrase of length zero occurs when a rule specifies that nothing is to be generated corresponding to the current element.)

Consider the following CD as input to the generator:

```
(S-HUNGER (ACTOR JOHN))
```

The generator first looks up the attributes of the first element of the CD, which in this case is the goal S-HUNGER. S-HUNGER has the following attributes:

```
S-HUNGER
--------
  GEN-SLOTS: '(ACTOR GOAL)
  PHRASE:    '("WAS HUNGRY")
  INFIN:     '("HUNGRY")
```

The first of these three attributes specifies the slots of the CD that are to be generated: in this case, the ACTOR and GOAL slots. (The "GOAL" slot denotes the first element in the CD, the CAR. Similarly, we will see that the "ACT" slot denotes the same element in an event CD, as opposed to a goal CD like the current example.)

These two GEN-GOAL slots are now considered, one at a time. The ACTOR slot filler is "JOHN". The relevant attributes under "JOHN" are as follows:

```
JOHN
----
  PHRASE:     ("JOHN")
  TYPE:       (HUMAN)
  GENDER:     (MALE)
```

The generator uses the "PHRASE" attribute to generate the first occurrance of JOHN. Based on the GENDER (MALE) attribute, the second occurrance of JOHN may be generated using the pronoun "HE", depending on two other tests: (1) how recently John has been mentioned and (2) whether or not any other male humans have been mentioned. The generator at this point outputs the English word "JOHN".

The second GEN-SLOT specifies the "GOAL" slot, which is the goal S-HUNGER. We just saw that the PHRASE attribute of S-HUNGER is the phrase "WAS HUNGRY". The generator simply generates this phrase, and then is finished. The entire sentence generated was "JOHN WAS HUNGRY".

The above was a very simple example, designed to show the basic processing. Complications set in when the context alters the meaning of a simple CD, and the generator must take this into account. For example, consider the following:

```
(ATRANS (ACTOR JOHN)
        (OBJECT MONEY)
        (TO MARY)
        (FROM JOHN))
```

This ATRANS action is generated in a manner similar to the S-HUNGER goal in the previous example. However, ATRANS can be generated as the English words "GOT" or "GAVE" depending on whether the ACTOR and TO slots contain the same slotfiller. Similarly, if ACTOR and TO are the same, then the FROM slot should be generated ("John got a book from the library"), but if those two slots differ, then the TO slot should be mentioned ("John gave a book to charity"). Hence, both the PHRASE and the GEN-SLOTS attributes of ATRANS contain rules for specifying the phrase and slots to generate, as follows:

```
ATRANS
------
  GEN-SLOTS: (COND ((EQUAL (ACTOR-SLOT *CD*)
                           (TO-SLOT *CD*))
                    '(ACTOR ACT OBJECT FROM))
                   (T
                    '(ACTOR ACT OBJECT TO)))
  PHRASE:    (COND ((EQUAL (ACTOR-SLOT *CD*)
                           (TO-SLOT *CD*))
                    '("GOT"))
                   (T
                    '("GAVE")))
```

In the current example, the ACTOR and TO slots are not the same: one contains JOHN and the other contains MARY. Hence, the slots to be generated are ACTOR, ACT, OBJECT and TO. The filler of the ACTOR slot causes "JOHN" to be generated. the ATRANS filler of the ACT slot causes "GAVE" to be generated (again, since the ACTOR and TO slots are different). The filler of the OBJECT slot causes the phrase "SOME MONEY" to be generated. The TO slot itself causes the word "TO" to be generated, and its filler generates "MARY".

Hence the entire sentence is "JOHN GAVE SOME MONEY TO MARY".

Now consider the following slightly different ATRANS:

```
(ATRANS (ACTOR JOHN)
        (OBJECT MONEY)
        (TO JOHN)
        (FROM MARY)
        (BUI (THREATEN OVERPOWER STEAL)))
```

This CD contains a BUI slot, which specifies the bottom-up index from this action, as described in Chapters 3 and 4. This particular bottom-up index specifies that John's ATRANS was part of either a THREATEN, OVERPOWER or STEAL plan. This information requires the generator to use a different word to specify the action. "GOT" does not adequately convey the action that John performed. Instead, the word "ROBBED" should be used.

Rather than specifying different phrases to use for every possible plan context that an action can appear in, ARTHUR instead stores that information under the indices themselves. Whenever an action contains a BUI slot, the PHRASE attribute of the default entry in that slot is looked up, instead of the PHRASE attribute under the action itself. Hence, in this example the ATRANS phrase is not accessed, rather, THREATEN is used. The following shows part of the PHRASE information specified under the THREATEN index:

```
THREATEN
--------
  PHRASE:    (COND ((EQUAL (ACT-SLOT *CD*)
                           '(ATRANS))
                    '("ROBBED")))
```

Hence, the above ATRANS example would be generated as "JOHN ROBBED SOME MONEY FROM MARY".

When the generator is operating in "question answering" mode (see Section 6.6.1), then some additional rules come into play. We saw that the Q/A module returns a list of goals in cases where the explanation for an action has been supplanted. This list constitutes a "history" of the goals that ARTHUR believed to be the explanations of the action. For example, recall the following:

[78] John went to the gas station. He robbed it.

When ARTHUR reads this story, it supplants the initial A-POSSESS (GAS) explanation by an A-POSSESS (MONEY)

explanation. When ARTHUR is asked "Why did John go to the gas station?", the answer is in the form of the two explanation triples which correspond to these two goals. These two explanation triples are passed to the generator, one at a time, and in addition, for each one, a "canned" phrase is generated, according to the following algorithm:

For the first goal in a list of supplanted goals,
insert the phrase ("AT FIRST I THOUGHT IT WAS BECAUSE")

For each subsequent goal in the list, until the final goal,
insert the phrase ("THEN I THOUGHT IT WAS BECAUSE")

For the final goal, which is active, not supplanted,
insert the phrase ("BUT ACTUALLY IT'S BECAUSE")

When ARTHUR generates the list of two goals A-POSSESS (GAS) and A-POSSESS (MONEY), it generates the above canned phrases interspersed between the generation of the goals themselves.

6.7 Summary

ARTHUR's processing of each new statement in a story consists of incorporating that statement into the story representation. The processing differs depending on the type of input read. ARTHUR recognizes three types of input: events, goals and new inference rules. The processing of an event consists of generating a bottom-up index, and then performing specification search to connect the event to a goal. When specification search fails to find a connection, then the initial goal may be connected to a new default goal via a goal relationship, or the initial goal may be supplanted by the new goal. If supplanting fails, then ARTHUR generates an improbable assumption, embodying a new inference rule which serves to connect the previously unconnected action and goal.

CHAPTER 7

EXTENDED EXAMPLES

7.1 Introduction

The previous chapters have outlined ARTHUR's processing algorithms, typically using very short two- or three-sentence illustrative examples. This chapter outlines some of the issues involved in ARTHUR's processing of some stories that are considerably longer and more realistic than the brief examples so far employed. Of the three stories examined, two are "real" stories, i.e., stories which were not written explicitly to fit ARTHUR's understanding theories. Understanding these stories nonetheless requires the use of adaptive understanding techniques. We will see that ARTHUR's understanding processes can account for the processing of these stories.

The third story was constructed for the purpose of illustrating all three of ARTHUR's major understanding processes in a single story. That story, entitled "A Weapon of War", requires the reader to be able to specify inference paths, supplant inferences and add new inference rules. Some of the actual annotated output from ARTHUR's processing of this story is presented, in order to demonstrate a view of ARTHUR's operation as an integrated system.

7.2 "The Count and the Wedding Guest"

The author William Sidney Porter was well known for writing stories with surprise endings, under the pseudonym of O. Henry. Saying that a story has a surprise ending implies that statements at the end of the story will cause a reader of the story to change one or more of the inferences he has already made while understanding the story. That is, a story with a surprise ending is intentionally misleading: it contradicts its own initial implications. A story of O. Henry's entitled "The Count and the Wedding Guest" is just such a story. The story is too long to be given in its entirety here, but the following is a brief synopsis of the

plot:

> A man named Andy Donovan meets a woman named Maggie Conway. Maggie wears a black mourning costume, and tells Andy that she mourns the death of her fiance, the Count Mazzini. She shows Andy a picture of the Count, which she carries in a locket. Andy woos her, and they soon become engaged. Then one night Maggie notices that Andy is upset. When she questions him, he explains that he knows an important politico named Big Mike Sullivan who he'd very much like to invite to their wedding. He tells Maggie that there's a reason why he can't invite Big Mike to their wedding, but that he can't tell her the reason.
>
> Maggie then suddenly confesses that she has a secret to tell him, at the risk of incurring his anger: there is no Count Mazzini; she made him up. She was never engaged to anyone, and she invented the whole thing because she had observed that men seemed to prefer ladies who had already had some previous serious relationship with another man. At this revelation, Andy immediately cheers up, and assures Maggie that he isn't angry, and that she's made everything alright. She asks him if he was fooled by her story. No, he replies, because the picture in her locket is a picture of Big Mike Sullivan.

There are a number of difficult issues that arise in understanding this story. We will concentrate first of all on the problem of understanding the surprise ending in the story. There are actually three surprises at the end of the story, i.e., three facts which contradict previous implications in the story. They are as follows:

(1) Maggie lied about having been engaged to the Count Mazzini,

(2) Andy recognized the picture in Maggie's locket,

(3) Andy knew Maggie had lied.

Understanding the story requires recognizing each of these three facts. The first of these is an example of a story character having been intentionally misleading. Hence

understanding (1) requires the reader to have knowledge about the truth or falsehood of explicit statements made by a story character.

In this case recognizing that Maggie had lied is not difficult, because she explicitly admits it. This means that our initial inference of her goal in wearing black and acting mournful must have been erroneous. Since there was no Count Mazzini, mourning for him could not be the explanation for Maggie's actions. Yet we initially believed that that was the explanation. Hence, understanding the story requires that this initial inference be supplanted by a new explanation, namely that Maggie wanted Andy to think that she had been engaged. If a reader were asked:

Q1) Why did Maggie wear a black mourning costume?

an appropriate answer would be:

A1) At first I thought it was because she was in mourning for the Count Mazzini, but actually it's because she wanted to make Andy believe that she had been engaged to another man.

This answer reflects the fact that the reader supplanted an initial erroneous inference with a corrected inference.

Chapter 4 presented a process by which an understander could recognize an erroneous inference and supplant it with a corrected inference. The rules presented in that chapter can be applied to the understanding of this story. Following is a brief outline of the steps involved in understanding the surprise ending in this story, and the relationship of these steps to the applicable rules for supplanting inferences presented in Chapter 4.

Step 1: (Rule 30: Specification failure)

Maggie's admission fails to be explained by her previously inferred goal of wanting to mourn her dead fiance.

Step 2: (Rule 33: Disconnected explanations)

Two separate goals have been stated for Maggie, which are not related by any known goal relationship.

Step 3: (Rule 35: Undoing a goal inference)

The earlier of the two goal explanations is assumed to have been inferred erroneously.

Step 4: (Rule 37: Supplanting a goal inference)

The action of wearing black, which had previously been explained as being in service of Maggie's goal of mourning her fiance, is now re-explained as being in service of her goal of making Andy believe that she had once been engaged.

Similarly, the other two surprises contained in the ending of this story may be viewed in terms of the process of supplanting an inference. Surprise (2) above was the fact that Andy recognized the picture in Maggie's locket. When this fact is stated, the reader recognizes that it contradicts a previous inference that Andy had believed Maggie's story that it was a picture of the Count Mazzini. Once that initial inference has been supplanted, then (3) above is stated: the fact that Andy knew that Maggie had lied. When Andy admits this the reader is forced to supplant the previous inference that Andy had believed Maggie. If the reader were asked:

Q1) Did Andy believe that Maggie had a dead fiance?

an appropriate answer would be:

A1) At first I thought he did believe her, but actually he didn't, because he recognized that the picture that she showed him was of someone he knew.

Again, this answer reflects the fact that the reader supplanted an initial erroneous inference with a corrected inference.

Understanding the surprising turn of events in "The Count and the Wedding Guest" requires the reader to have made some tentative inferences about the story characters' beliefs and intentions, and then to have supplanted those inferences on the basis of the facts revealed at the end of the story. The rules for supplanting inferences given in Chapter 4 are applicable to the process of supplanting the inferences in this story.

7.3 "The War of the Ghosts"

"The War of the Ghosts" is an Eskimo folk tale which has been frequently referenced in the psychological literature, first by Frederic Bartlett [1932] in his early work on human memory. The story is as follows:

> One night two young men from Egulac went down to the river to hunt seals, and while they were there it became foggy and calm. Then they heard war-cries, and they thought: "Maybe this is a war-party". They escaped to the shore, and hid behind a log. Now canoes came up, and they heard the noise of paddles, and saw one canoe coming up to them. There were five men in the canoe, and they said:
>
> "What do you think? We wish to take you along. We are going up the river to make war on the people".
>
> One of the young men said: "I have no arrows".
>
> "Arrows are in the canoe", they said.
>
> "I will not go along. I might be killed. My relatives do not know where I have gone. But you", he said, turning to the other, "may go with them."
>
> So one of the young men went, but the other returned home.
>
> And the warriors went on up the river to a town on the other side of Kalama. The people came down to the water, and they began to fight, and many were killed. But presently the young man heard one of the warriors say: "Quick, let us go home: that Indian has been hit". Now he thought: "Oh, they are ghosts". He did not feel sick, but they said he had been shot.
>
> So the canoes went back to Egulac, and the young man went ashore to his house, and made a fire. And he told everybody and said: "Behold I accompanied the ghosts, and we went to fight. Many of our fellows were killed, and many of those who attacked us were killed. They said I was hit, and I did not feel sick".

> He told it all, and then he became quiet. When the sun rose he fell down. Something black came out of his mouth. His face became contorted. The people jumped up and cried.
>
> He was dead.

There are many difficult problems involved in understanding this story. We will not attempt to give an overview of the processing of the whole story, since it is beyond the scope of this thesis to do so. We will concentrate on one specific event in the story, and attempt to account for the understanding of that event.

One of the most striking events in this story is that of "something black" coming out of the warrior's mouth just before he dies. It seems natural to associate the event of the black expulsion with the warrior's death. However, that event is one which was surely unknown to the reader before reading this story. Hence, the reader could not have had any previous knowledge of what inferences to make from such an event. The problem is, how can the reader recognize any connection between this event and the warrior's death?

The process of forcing an interpretation is a method of adding new facts about the world to memory so as to allow previously unknown inferences to be generated. In this example, the event of a person expelling something black from his mouth is stated next to the event of that person dying. We hypothesize that the expulsion is a novel event to the reader. Hence, initially, the reader will be unable to find any known connection between the two events. Since no default explanation could have been made from the "expel" event, there is no erroneous inference to supplant. Hence, the reader will attempt to force an interpretation from the stated "death" state-change explanation onto the as yet unexplained expel event.

Rule 52 from Chapter 5 says that when forcing an interpretation, the default top-down index from the explanation is assumed to be a bottom-up index from the action. The default top-down index from the "death" explanation is the CHANGE-HEALTH (NEG) index. Hence, in this case, the CHANGE-HEALTH index is now assumed to be the bottom-up index from the "expel" event. This means that the understander adds CHANGE-HEALTH as a new bottom-up index to the action of EXPELling a black object from one's mouth. This new inference can be stated in the form of the following rule:

> If a character EXPELs something black from his mouth,
>
> Then infer that that character may become dead (HEALTH VAL (-10)).

Our theory predicts that the reader will add this new inference rule to his memory. Hence, we also predict that if a reader were asked:

> Q1) What would happen if something black came out of another warrior's mouth?

then the reader would answer:

> A1) Then that warrior would probably die too.

Understanding this story requires the reader to account for this anomolous "expel" event somehow. Our theory says that this explanation will be effected by the process of forcing an interpretation, which will result in the construction of the new inference rule given above, according to the process described in Chapter 5. This can account for the fact that readers associate the event with the warrior's death. It furthermore predicts that from that point on, (within the context of this story), the reader will continue to infer that this expel event leads to death, until that inference is contradicted by some other information in the story.

7.4 "A Weapon of War": A Detailed Example

Consider the following story:

[79] "A Weapon of War"

> The war between the Empire and the Rebel army had been raging for nearly a year. The Empire's evil King Raoul ordered Eric the inventor to come up with a new weapon. Eric presented Raoul with a newly developed weapon. Eric said: "If the button on the weapon is touched, and a person approaches the weapon, then that person will die." Eric then bowed low and left the room. Raoul sat and smiled for many minutes. Then he pressed the button on the weapon, and he spoke into his intercom. "Send in my wife," he said.

This story is intentionally misleading. It contains a surprise ending: Raoul intends to use the weapon to kill his wife, not to kill Rebel soldiers. Understanding the story involves our changing our minds about Raoul's goal in two of his previous actions: first, acquiring the war weapon, and second, pushing the button to activate the weapon. ARTHUR implements this process by supplanting an initial "win war" goal with a "kill wife" goal.

Furthermore, the story describes a new object, with certain specified properties: a weapon, which requires a button's depression and a victim's approach, as steps in a plan to decrease the victim's health to the level of death. Understanding the story requires recognizing that Raoul's actions of pushing the button and ordering his wife in are both part of the use of the new device. ARTHUR incorporates this new information about the weapon in the form of a new inference rule. The inference rule is added to ARTHUR's memory, and then is used to give rise to new and crucial inferences from what would otherwise be irrelevant actions.

ARTHUR has successfully understood this story, input directly in CD. The English in this story cannot be parsed by ARTHUR. The rest of this section presents a brief sentence-by-sentence analysis of ARTHUR's processing of this story, pointing out some of the issues and difficulties that arise in the process.

[1] The war between the Empire and the Rebel army had been raging for nearly a year.

This sentence contains a statement of two goals: an ACHIEVE-WIN of a war, by both the Empire and the Rebels. The word "war" conveys the information that the two goals are in a state of goal competition with each other.

[2] The Empire's evil King Raoul ordered Eric the inventor to come up with a new weapon.

Since a weapon has a known function of either THREATENing or OVERPOWERing someone, or causing someone's health to go down, King Raoul's order can be interpreted as part of a plan to use the new weapon to THREATEN, OVERPOWER or CHANGE-HEALTH (NEG) of some victim. Since Raoul is stated to be part of the social entity "Empire", we can infer that he shares the Empire's ACHIEVE-WIN (WAR) goal against the Rebels; that is, he has an identical A-WIN goal in concord with the Empire's goal. Hence, the above three plans can be inferred to be directed against the Rebels as

victims, in service of Raoul's presumed goal of winning the war.

[3] Eric presented Raoul with a newly developed weapon.

Eric's ATRANS is inferred to be in compliance with Raoul's previous order. This causes us to infer still another ACHIEVE-WIN goal: Eric too wants to win the war against the Rebels, again in concord with the Empire's and Raoul's goals.

[4] Eric said: "If the button on the weapon is touched, and a person approaches the weapon, then that person will die."

This is a statement of a new inference rule: Eric identifies two actions that can now be interpreted as previously unknown steps in a plan to cause a person's death. The steps are (1) coming into physical contact with the button, (2) a person's coming into proximity of the weapon and (3) the person's death. The first two steps are represented by the indices PHYS-CONTACT and DPROX. The new inference rule states that these two indices can both be considered to be steps in a plan to cause someone to die (CHANGE-HEALTH (-10)). Hence, from now on the understander will recognize that these two indices can be interpreted as being part of this plan.

[5] Eric bowed low and left the room.

Eric's action can only be explained as being part of a social norm of behavior: having achieved the purpose of his meeting with Raoul, namely, giving him the war weapon, he is then supposed to leave. This information is not known to ARTHUR, hence, ARTHUR would not recognize this explanation for the action. Rather, it would leave a loose-end explanation for a PTRANS to an unspecified location. Thus if ARTHUR were asked why Eric left, it would answer that it didn't know why. A human understander, using the situational knowledge described above, could answer that Eric might have left because he had finished his meeting with Raoul.

[6] Raoul sat and smiled for many minutes.

Again, a statement that adds little to our understanding of the story. We can tell because this statement can be left out, without affecting the overall meaning of the story, as evidenced by our answers to questions about the events and intentions in the story. The statement can be explained in terms of Raoul's satisfaction at the fulfillment of his previously stated intention to get a new war weapon.

[7] Then he pressed the button on the weapon, and he spoke into his intercom.

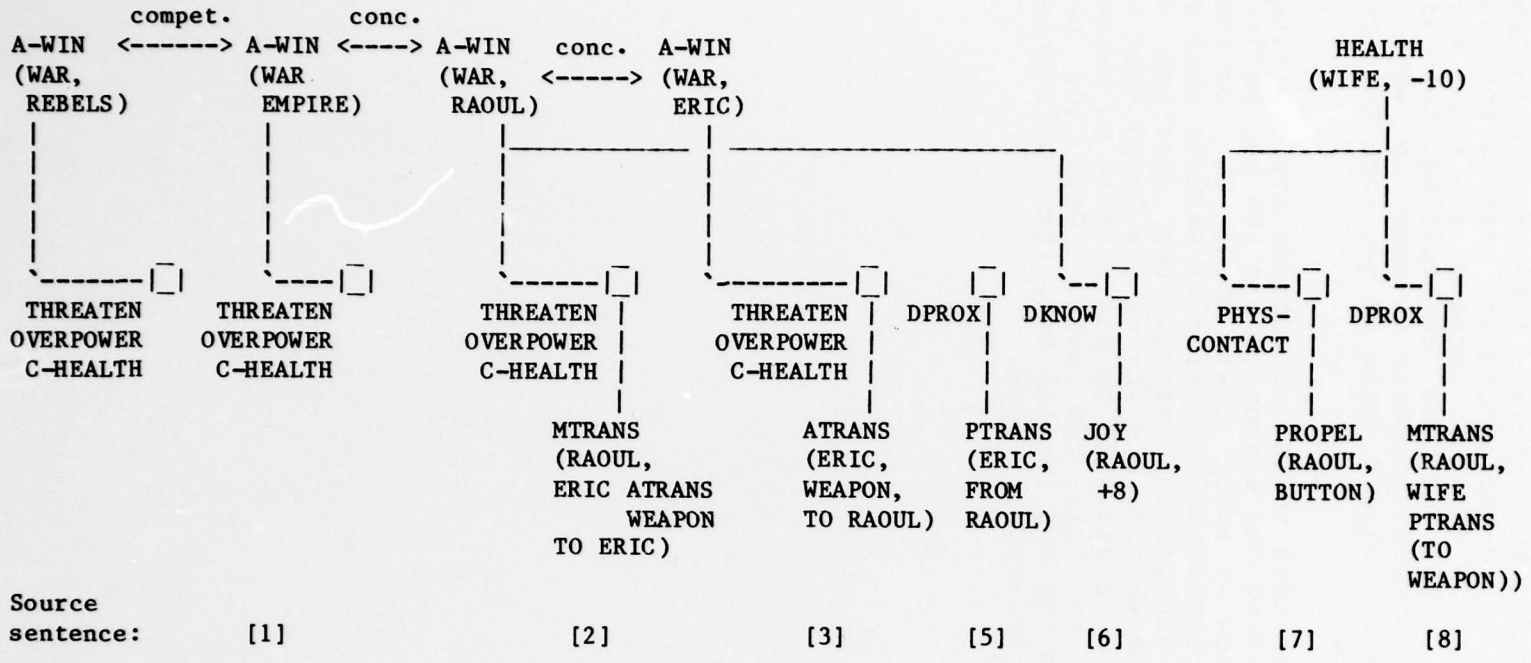
This sentence contains two statements, a PROPEL BUTTON and an MTRANS. In the first statement, Raoul has performed one of the actions which was specified as being part of a plan of killing someone, in sentence [4]. Hence the default inference from this action is that Raoul intends to kill someone with the device. We don't yet know who his intended victim is.

The action of speaking into the intercom is an MTRANS with no MOBJECT or RECIPIENT slot specified. Hence by Rule 16 in Chapter 2, ARTHUR leaves a loose-end explanation for the action. That is, the action is not specified enough to permit the inference of a default explanation. Hence ARTHUR instead awaits further specification of the action.

[8] "Send in my wife," he said.

This statement specifies the previous loose-end MTRANS. The MOBJECT here is a statement of an order for Raoul's wife to PTRANS herself to Raoul's location. The understander must be able to recognize that this is also the location of the war weapon. One of the bottom-up indices from a PTRANS is a DPROX, which has been augmented by the new inference rule in sentence [4] to be part of a plan of killing someone. The other step in that plan, pushing the weapon's button, has already been effected. Hence the default inference from this action is that Raoul's wife will get killed by approaching the weapon. Since Raoul himself was the planner of both of the steps in this plan, the default explanation is that Raoul wants to cause his wife's death.

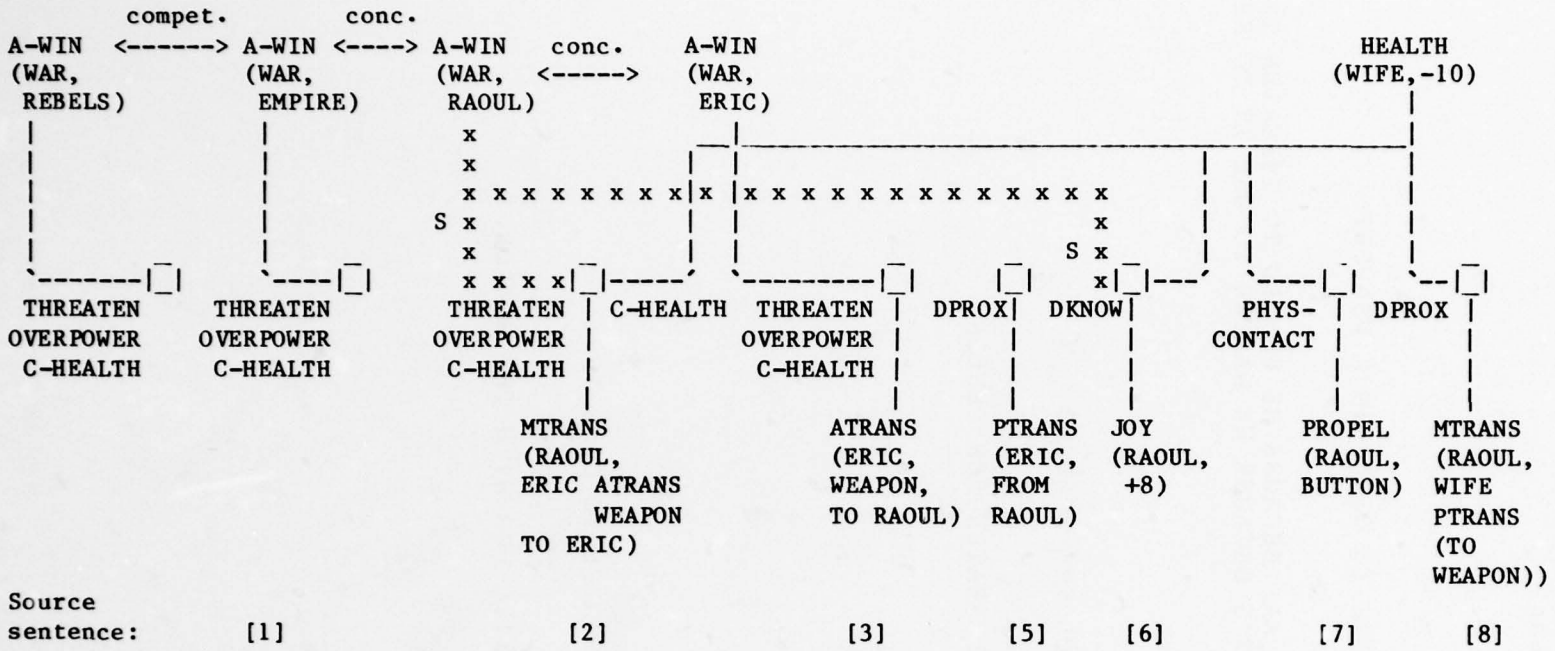
Having inferred this, the understander then has a disconnected explanation: some of Raoul's actions are explained as being in service of a goal of winning a war against the Rebels, and some are explained as being in service of this newly inferred goal of killing his wife. The following figure contains a sketchy version of the explanation-graph representation for the disconnected explanation that exists at this point in the story. (Note that Sentence [4] is not represented in this explanation graph, because it is a statement of a new inference rule, which is not representable in an explanation graph.)


compet.
conc.
A-WIN
(WAR,
REBELS)
A-WIN
(WAR
EMPIRE)
A-WIN
(WAR,
RAOUL)
conc.
A-WIN
(WAR,
ERIC)
HEALTH
(WIFE, -10)
THREATEN
OVERPOWER
C-HEALTH
THREATEN
OVERPOWER
C-HEALTH
THREATEN
OVERPOWER
C-HEALTH
THREATEN
OVERPOWER
C-HEALTH
DPROX
DKNOW
PHYS-
CONTACT
DPROX
MTRANS
(RAOUL,
ERIC ATRANS
WEAPON
TO ERIC)
ATRANS
(ERIC,
WEAPON,
TO RAOUL)
PTRANS
(ERIC,
FROM
RAOUL)
JOY
(RAOUL,
+8)
PROPEL
(RAOUL,
BUTTON)
MTRANS
(RAOUL,
WIFE
PTRANS
(TO
WEAPON))
Source
sentence:
[1]
[2]
[3]
[5]
[6]
[7]
[8]

[Figure 24]

According to Rules 35 and 37 in Chapter 4, the understander will now attempt to create a connected explanation by supplanting the initial inference with the new one. That requires re-explaining any actions that had been explained by the A-WIN goal, in terms of the new "kill wife" (HEALTH) goal. The actions re-explained here are Raoul's initial action of ordering Eric to come up with a new weapon, and his subsequent smile of joy once Eric has complied with his order.

We initially infer that these two actions are explained by Raoul's goal of using the weapon to win the war against the Rebels. Hoever, now we infer that these two actions actually should be explained in terms of Raoul's goal of killing his wife with the weapon. Hence, the question of why Raoul asked for the weapon, and why he was happy when he got it, should both now be answered by saying that he wanted to use the weapon to kill his wife. Once the new explanation has supplanted the initial one, the explanation graph looks as in the following figure.


compet.
conc.
A-WIN
(WAR,
REBELS)
A-WIN
(WAR,
EMPIRE)
A-WIN
(WAR,
RAOUL)
conc.
A-WIN
(WAR,
ERIC)
HEALTH
(WIFE,-10)
THREATEN
OVERPOWER
C-HEALTH
THREATEN
OVERPOWER
C-HEALTH
THREATEN
OVERPOWER
C-HEALTH
C-HEALTH
THREATEN
OVERPOWER
C-HEALTH
DPROX
DKNOW
PHYS-
CONTACT
DPROX
MTRANS
(RAOUL,
ERIC ATRANS
WEAPON
TO ERIC)
ATRANS
(ERIC,
WEAPON,
TO RAOUL)
PTRANS
(ERIC,
FROM
RAOUL)
JOY
(RAOUL,
+8)
PROPEL
(RAOUL,
BUTTON)
MTRANS
(RAOUL,
WIFE
PTRANS
(TO
WEAPON))
Source
sentence:
[1]
[2]
[3]
[5]
[6]
[7]
[8]

[Figure 25]

Following is the input and output of a question-answering session with ARTHUR after it has understood "Weapon of War".

```
Input question:*WARQ1

QUESTION:
 WHY DID RAOUL ORDER ERIC TO GIVE HIM A WEAPON?

ANSWER:
 AT FIRST I THOUGHT IT WAS BECAUSE HE WANTED TO THREATEN THE
REBELS, BUT ACTUALLY IT'S BECAUSE HE WANTED TO USE IT TO KILL
HIS WIFE.

Input question:*WARQ2

QUESTION:
 WHY DID ERIC GIVE RAOUL THE WEAPON?

ANSWER:
 BECAUSE HE WANTED TO THREATEN THE REBELS.

Input question:*WARQ3

QUESTION:
 WHY DID ERIC LEAVE?

ANSWER:
 I DON'T KNOW.

Input question:*WARQ4

QUESTION:
 WHY DID RAOUL TELL HIS WIFE TO COME IN?

ANSWER:
 BECAUSE HE WANTED TO KILL HER WITH THE WEAPON.
```

7.5 Summary

This chapter has outlined some of the issues that arise in the processing of two real stories, "The Count and the Wedding Guest", and "The War of the Ghosts". The first story has a surprise ending, and hence requires understanders to be able to change their minds about their initial inferences, in order to account for the new information presented at the end of the story. Previous understanding programs would not be able to understand this story, because they are unable to change their minds about their own inferences. The rules underlying the process of supplanting an inference were shown to be useful in understanding this story.

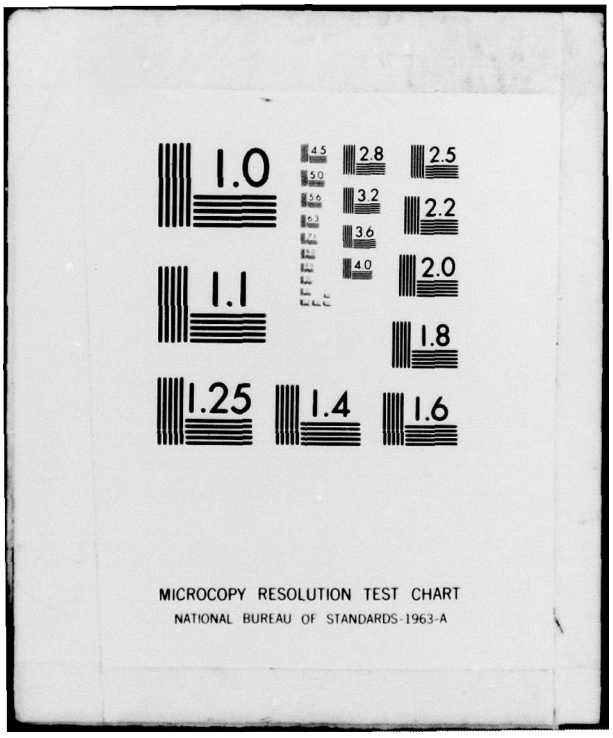
"The War of the Ghosts" presents the problem of recognizing a previously unknown connection between the action of a person expelling something black from his mouth and that person's subsequent death. That connection is generally recognized by human readers of the story. However, previous understanding systems would be unable to recognize this connection, if it was not pre-specified in the understander's memory. ARTHUR's ability to add facts to its memory in the form of new inference rules allows it to recognize this previously unknown connection, via the process of forcing an interpretation.

"Weapon of War" is a story that was constructed for the purpose of illustrating how ARTHUR's three processes of specifying inference paths, supplanting inferences and adding new inference rules could work together in an integrated system. ARTHUR employs all three processes at various times in understanding this story.

AD-A081 012 YALE UNIV NEW HAVEN CONN DEPT OF COMPUTER SCIENCE F/G 5/7
ADAPTIVE UNDERSTANDING: CORRECTING ERRONEOUS INFERENCES.(U)
JAN 80 R H GRANGER N00014-75-C-1111
UNCLASSIFIED RR-171 NL

3 OF 3
AD
A081012

END
DATE FILMED
3-80
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
7.1
8.0
9.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

BIBLIOGRAPHY

1) Abelson, R. P. (1973). The structure of belief systems. In R. C. Schank and K. M. Colby (eds.), Computer Models of Thought and Language. Freeman, San Francisco

2) Abelson, R. P. and Reich, C.M. (1969). Implicational molecules: a method for extracting meaning from input sentences. In D. E. and L. M. Norton (eds.), Proceedings of the First International Joint Conference on Artificial Intelligence:641-648.

3) Bartlett, R. (1932). Remembering: A Study in Experimental and Social Psychology. Cambridge University Press, London.

4) Becker, J. (1973). A Model for Encoding Experiential Information. In R. C. Schank and K. M. Colby, eds. Computer Models of Thought and Language. Freeman, San Francisco.

5) Birnbaum, L. and Selfridge, M. (1979). Problems in Conceptual Analysis of Natural Language. Yale University, Computer Science Dept., Research Report #168.

6) Bobrow, D. and Collins, A. (1976). Represenation and Understanding: Studies in Cognitive Science. Academic press, New York.

7) Carbonell, Jaime(1979). Subjective Understanding: Computer Models of Belief Systems. Yale University Research Report #150.

8) Carbonell J. G. (1978). "Politics: Automated Ideological Reasoning". COGNITIVE SCIENCE,(2), pp. 27-51.

9) Charniak, E. (1972). Towards a model of children's story comprehension. AI TR-266, MIT.

10) Charniak, E. (1974). He will make you take it back: A study in the pragmatics of language. Techncial Report, Istituto per gli studi semantici e Cognitivi, Castagnola, Switzerland.

11) Charniak, E. (1978). On the use of framed knowledge in language comprehension. (In press).

12) Collins, A. (1976). Processes in acquiring knowledge. In R. C. Anderson, R. J. Spiro, and W. E. Montague (eds.) Schooling and the acquisition of knowledge. Lawrence Erlbaum Associates, Hillsdale, New Jersey.

13) Conrad, C. (1972). Cognitive economy in semantic memory. Journal of Experimental Psychology, 92(2). Behavior 8.

14) Cullingford, R. (1977). Controlling Inferences in Story Understanding. Fifth International Joint Conference on Artificial Intelligence, Cambridge, MA.

15) Cullingford, R. E. (1978). Script Application: Computer Understanding of newspaper stories. Yale University Research Report #116.

16) DeJong, G.F. (1979). Skimming Stories in Real Time: An Experiment in Integrated Understanding. Ph.D. thesis, Dept. of Computer Science, Yale University, research report #158

17) Fahlman, S. (1974). A planning system for robot contruction tasks. Artificial Intelligence 5:1-50.

18) Feigenbaum, E. and Feldman, J. (1963). Computers and Thought. McGraw-Hill, Inc., New York.

19) Gershman, A.V. (1979). Knowledge-Based Parsing. Research Report #156, Computer Science Dept., Yale University, New Haven, Conn.

20) Goldman, N. (1973). The generation of English sentences from deep conceptual base. Ph.D. thesis, Computer Science Dept. Stanford University, Stanford, CA.

21) Goldman, N. (1975). Conceptual generation. In R. C. Schank, ed. Conceptual Information Processing. North Holland Publ, Amsterdam.

22) Granger R. (1977). FOUL-UP: A program that figures out meanings of words from context. Fifth International Joint Conference on Artificial Intelligence, Cambridge, MA.

23) Hemphill, L. (1973). The relationship of language and belief: With special emphasis on English 'for' constructions. Ph.D. thesis, Linguistics Dept. Stanford University.

24) Joshi, A. K. and Rosenschein, S. J. (1976). Some Problems of Inferencing: Relation of Inferencing to Decomposition of Predicates. In Proceedings of the International Conference on Computational Linguistics (COLING 1976), Ottawa, Canada.

25) Katz, J., and Fodor, J. (1963). The structure of a semantic theory. Language 39.

26) Kintsch, W. (1974). The representation of meaning in memory. Wiley, New York.

27) Kolodner, J. L. (1978). Memory organization for natural language database inquiry. Research Report No. 142, Computer Science Department, Yale University, New Haven, CT.

28) Lehnert, W. (1977). Human and computational question answering. Cognitive Science 1.

29) Lehnert, W. (1977). The Process of Question Answering. Lawrence Erlbaum Associates

30) Lesser, V. and Erman, L. (1977). A retrospective view of the Hearsay-II architecture. Fifth International Joint Conference on Artificial Intelligence, Cambridge, MA.

31) McDermott, D. V. (1974). Assimilation of new information by a natural language understanding system. MIT AI Laboratory TR-291.

32) Meehan, J. (1976). The Metanovel: Writing stories by computer. Ph. D. thesis, Yale University.

33) Minsky, M., ed. (1968). Semantic Information Processing. The MIT Press, Cambridge, Mass.

34) Minsky, M. (1974). A framework for representing knowledge. MIT. AI Memo No. 306.

35) Newell, A. and Simon, H. A. (1972). Human Problem Solving. Prentice Hall, Inc., New Jersey.

36) Norman, D. and Rumelhart, D. (1975). Explorations in cognition. W. H. Freeman, San Francisco.

37) Piaget, J. (1955). The Language and Thought of the Child. Meridian Books

38) Propp, V. (1970). Morphology of the Folktale. University of Texas, Austin.

39) Quillian, M. R. (1968). Semantic memory. In M. Minsky, ed. Semantic Information Processing. MIT Press, Cambridge.

40) Rieger, C. (1975). Conceptual memory. In R. C. Schank, ed. Conceptual Information Processing. North Holland, Amsterdam.

41) Schank, R. C. (1969). A conceptual dependency representation for a computer-oriented semantics. Ph. D. thesis, University of Texas.

42) Schank, R. C. (1973). Causality and reasoning. Technical Report #1, Istituto per gli Studi Semantici e Cognitivi, Castagnola, Switzerland.

43) Schank, R. C. (1973). Identification of conceptualizations underlying natural language. In R. C. Schank and K. Colby, eds. Computer Models of Thought and Language. W. H. Freeman, San Francisco.

44) Schank, R. C. (1975). Conceptual Information Processing. North Holland, Amsterdam.

45) Schank, R. C. (1975). The structure of episodes in memory. In D. Bobrow and T. Collins, eds. Representation and Understanding: Studies in Cognitive Science. Academic, New York.

46) Schank, R. C. and Abelson, R. P. (1977). Scripts, Plans, Goals, and Understanding. Lawrence Erlbaum Press, Hillsdale, N.J.

47) Schank, R. C. and Colby, K., editors (1973). Computer Models of Thought and Language. Freeman Press, San Fransisco, Calif.

48) Selfridge, O. G. (1959). Pandemonium: A Paradigm for Learning. Proceedings of the Symposium on Mechanism of Thought Processes. 2 vols. National Physical Laboratory, Teddington, England. London: H.M. Stationary Office, 1959, pp. 511-529.

49) Sussman, G. J., and McDermott, D. V. (1972). Why CONNIVING is better than PLANNING. MIT AI Memo No. 255A.

50) Tulving, E. and Donaldson, W. (eds.), (1972). Organization of memory. Academic Press, New York.

51) Waterman and Hayes-Roth, editors (1978). Pattern-directed Inference Systems. Academic Press, New York.

52) Wilensky, R. (1978). Understanding Goal-based Stories. Research Report #140, Yale University, Department of Computer Science, New Haven CT.

53) Wilks, Y. (1975). A preferential pattern-seeking, semantics for natural language inference. Artificial Intelligence 6, 53-74.

54) Winograd, T. (1972). Understanding Natural Language. Academic Press, New York.

55) Woods, W. A. (1970). Transition network grammars for natural language analysis. Communications of the Association for Computing Machinery, Vol. 13, No. 10.

56) Woods, W., Kaplan, R., and Nash-Webber, B. (1972). The lunar sciences natural language information system: final report. Bolt, Beranek and Newman Report No. 2378, Cambridge, MA. language analysis. Comm. of the ACM 13(10):591-606.